A SELF-ADAPTIVE COMPUTATIONAL METHOD FOR TRANSONIC TURBULENT PROJECTILE AERODYNAMICS

BY

CHRISTOPHER WILLIAM REED

A DISSERTATION PRESENTED TO THE GRADUATE SCHOOL
OF THE UNIVERSITY OF FLORIDA IN PARTIAL FULFILLMENT
OF THE REQUIREMENTS FOR THE DEGREE OF DOCTOR OF PHILOSOPHY

UNIVERSITY OF FLORIDA

1987

To Shelley and Drew

ACKNOWLEDGEMENTS

The support of Dr. Chen-Chi Hsu has been most encouraging during the course of this work. His guidance and willingness to provide it are appreciated. In giving me the freedom to experiment, discover, fail and succeed I have learned as much about research as I have about the topic. For this I am grateful.

My appreciation is extended as well to the committee members, Dr. Leadon, Dr. Kurzweg, Dr. Mikolaitis and Dr. Wilson.

This work was partially supported by the AFOSR. Computer time was provided by NASA Ames, NSF and the Pittsburgh Supercomputer Center.

TABLE OF CONTENTS

Abstract of Dissertation Presented to the Graduate School
of the University of Florida in Partial Fulfillment of the
Requirements for the Degree of Doctor of Philosophy

A SELF-ADAPTIVE COMPUTATIONAL
METHOD FOR TRANSONIC TURBULENT
PROJECTILE AERODYNAMICS

By

Christopher William Reed

August, 1987

Chairman: Chen-Chi Hsu
Major Department: Engineering Sciences

An adaptive grid generation procedure is developed for viscous flow problems. The equations governing the adaptation are based on a variational statement resulting in a set of elliptic equations in which adaptation can occur independently in each coordinate direction. The method allows for explicit control of adaptation and orthogonality while grid smoothness is implicit in the elliptic equations. The adaptive grid generation equations retain a simple relationship between the control functions and the grid point spacing, are capable of efficiently providing the extremely refined mesh in the boundary layer regions and can maintain a smooth grid network in complex geometries.

A self-adaptive computational technique has been developed by coupling the adaptive grid generation procedure with a thin layer Navier-Stokes code and a TVD code. In solving an axisymmetric turbulent

transonic projectile flow problem, the grid network was adapted to the pressure distribution in the streamwise direction and to the velocity distribution in the direction normal to the projectile surface. Results obtained for Mach 0.91, 0.96 and 1.2 indicate that the procedure can provide good adapted grid networks provided good choices are made for the control functions.

CHAPTER I

INTRODUCTION

The study of boundary fitted grid systems has become an important part of computational fluid dynamics and, in fact, their development has been cited as a major pacing item in the progress of computational fluid dynamics [1]. The use of such systems makes the application of finite difference techniques versatile and effective in solving complex fluid dynamic problems in arbitrary domains. Boundary fitted grid generation can be viewed as the development of a general curvilinear coordinate system in which the coordinates coincide with the boundaries on the edges of the physical domain. This feature of the boundary fitted grid systems facilitates the application of finite difference approximations at the boundaries and eliminates the need for interpolation between grid points near the boundary. This is particularly advantageous when the boundaries are highly curved or have slope discontinuities since the use of interpolation near these boundaries may have a significant effect on the solution [2].

In the domain interior, the intersection of each family of coordinate lines denotes each grid point and therefore defines the computational mesh. The distribution of the coordinate lines in the physical domain may

be concentrated to provide high resolution in certain areas but will always form an equally spaced rectilinear coordinate system in the transformed plane. Thus, once the governing equations are transformed onto the curvilinear coordinates the finite difference algorithms used to solve these equations may be developed on the equally spaced rectilinear coordinate system which is inherent in the transformed plane. Certain characteristics of the general curvilinear coordinate system have been shown to affect the accuracy of the finite difference approximations in the transformed plane [3]. These include orthogonality of intersecting coordinate lines and the smoothness of the grid spacing. It is also known that a refined mesh is needed in regions of large physical gradients to obtain accurate approximations. In fact, poor resolution in these high gradient regions can be detrimental to solution accuracy and to the convergence of finite difference algorithms for fluid flow problems [4]. Thus, the development of a good adapted grid network is important for the successful application of finite difference methods to solve complex fluid dynamic problems.

The self-adaptive grid generation technique has become an important area in computational fluid dynamics since it has been shown to provide good grid networks for the complex flow fields occurring in transonic and supersonic flows [5,6,7]. The use of boundary fitted curvilinear coordinate systems with transformed governing equations leads naturally to the concept of solution adaptive grids. As constraints of computer storage and CPU time prohibit uniform mesh refinement throughout the entire domain, the coordinate spacing in the physical domain is varied to increase resolution in only those regions in which the solution is changing rapidly. For simple flow problems, when the position of

important solution gradients is known, good adapted grids can be obtained with conventional techniques. However, in more complex problems the position of these important regions is not known a priori and then the development of a proper adaptive grid network is difficult. Solution adaptive grid generation adresses this problem by continuously updating the grid network as the solution evolves such that the important physical gradients are sufficiently resolved as they develop. It is thus an attempt to efficiently reduce the truncation error by only refining the mesh in those regions where the error may be large. Also, since the grid characteristics of smoothness and orthogonality also affect the truncation error, a good adaptive grid generation technique should include the enhancement of these characteristics as well.

The general approach of most adaptive grid generation schemes is based on minimization techniques. A measure of each desired grid characteristic is defined, and the governing equations for the grid are obtained by minimizing the integral of these measures over the domain. In many instances adaptation is required in only one coordinate direction. Dwyer et al. [8], for instance, have developed an equidistribution law to control the grid spacing along one coordinate direction. The spacing was adapted to a flame front in a two-dimensional combustion problem and the improved resolution eliminated spacial oscillations in the flame front. Gnoffo [9] has developed another one dimensional adaptive scheme in which tension springs are assumed to connect the grid points along a coordinate. The spring constants became large when increased resolution was required, and the points clustered in those regions in an attempt to minimize the potential energy of the spring system. The application of this technique to supersonic flow over a Viking Aeroshell successfully

resolved a separated shear layer by adapting to the velocity gradient.

One dimensional adaptation can be extended to higher dimensions by successive adaptation in each coordinate direction. Nakahashi and Deiwert [7] have used the spring analogy in such an approach and added torsional springs that connect intersecting coordinate lines to control orthogonality. They solved a transonic viscous airfoil problem in which the grid was adapted to the density gradient in the streamwise direction to resolve shocks and adapted to the velocity gradient normal to the airfoil surface to resolve the boundary layer. The one-dimensional approach to multi-dimension adaptation has the advantage of efficiency since the grid can usually be obtained using a marching solution algorithm. Another advantage is the independence of adaptation in each coordinate direction. However, it is difficult to maintain smoothness in such approaches, and highly skewed grids may result [10].

Rai and Anderson [11] have used a different approach for two dimensional adaptation. In their scheme each grid point induces a velocity, either repulsive or attractive, on every other point. By choosing some flow gradient to indicate the need for point clustering, each point with a gradient larger than the average gradient attracts points, and those with values below the average repel points. By summing the induced velocities from each point, the velocity of each point is determined and can be integrated over a time step to determine the new point location. In applying this scheme to the two dimensional linearized viscous Burger's equation, they adapted the grid to the derivative of the dependent variable in each direction and showed that such an approach will increase solution accuracy.

Saltzman and Brackbill [5] have developed an adaptive grid generation scheme based on a variational approach. A functional is defined to measure each grid characteristic and the minimization of these functionals results in a set of partial differential equations that govern the grid point spacing. With the proper choice of parameters the equations are elliptic, which guarantees a one-to-one mapping and results in a smooth grid. They solved an inviscid supersonic flow problem by adapting the grid cell size to a function of the pressure gradient to capture shocks. However, it is not possible in this approach to adapt the grid spacing independently in each coordinate direction, a feature that is prohibitive if the spacing in different coordinate directions vary by a large amount.

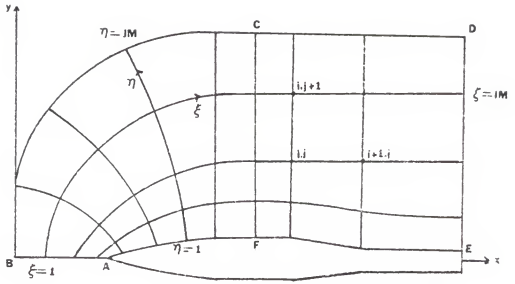
The adaptive grid generation technique presented here is similar in approach to that of Brackbill and Saltzman but is developed for applications to viscous transonic flow problems. The solution of these problems usually contains shock structures aligned with one coordinate direction and boundary shear layers parallel to a streamwise coordinate. Also, the grid point spacing can vary by orders of magnitude along different coordinates and it is thus necessary to adapt the grid independently in each coordinate direction. Therefore, a functional is defined for independent adaptation in each coordinate direction such that the resulting equations are elliptic. The use of elliptic equations in grid generation offer some important advantages that motivate their use. They constitute a boundary value problem and can thus be used in any domain. They guarantee a one to one mapping and also, grid smoothness is inherent in the equations. In addition to the equations governing the grid, there are two other topics that require attention when developing

an adaptive grid generation technique. First is the definition of the control functions, which are the part of the grid generation equations that dictate the need for adaptation. They are dependent upon the solution and thus provide the link between the grid generation and the governing equations. Also, a method is needed to account for the effect of the moving grid on the solution. Since the solution is defined at each point, any movement will alter the solution. The proposed adaptive grid generation technique is used to adapt the grid for a transonic axisymmetric projectile flow problem which is solved using both an implicit factorized algorithm and a TVD scheme for the thin layer Navier Stokes equations. The constraint of axisymmetric flow reduces the domain to two independent spatial variables, thus the grid generation equations are developed in two dimensions. The technique, however, can be readily extended to three dimensions.

CHAPTER II
THE ADAPTIVE GRID GENERATION EQUATIONS

II.1 The Curvilinear Coordinate System

The use of boundary fitted curvilinear coordinate systems in solving equations on arbitrarily shaped domains was originally motivated due to problems in applying finite difference approximations at curved boundaries. Because curved boundaries do not in general coincide with the rectilinear Cartesian coordinates, the grid points defined along these coordinates do not usually lie on the bounding curve. It therefore becomes difficult to accurately represent the boundaries [2] and special procedures must be used in those regions [12]. A solution to this problem, which has become an effective and versatile approach, is to map the arbitrarily shaped physical domain (x,y) into a rectangular computational domain (ξ,η). Such a mapping for a typical projectile flow domain is shown in Figure 2.1. As indicated, the curved ξ and η coordinate lines coincident with the boundaries in the physical domain map onto the edges of the rectangular computational domain. The boundary grid points, being defined by the coordinate intersections, are coincident with the boundary in the computational plane and thus no special treatment for the boundary conditions is required. Another advantage is that virtually any set of nonuniformally spaced curvilinear coordinates in the physical domain map into an equally spaced regular



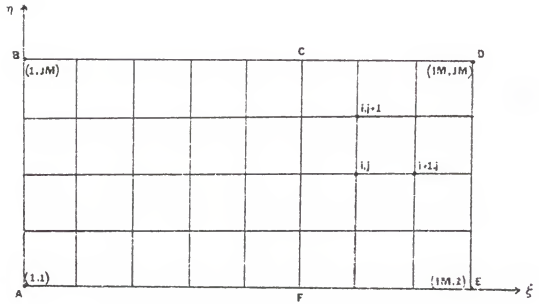


Figure 2.1 Curvilinear coordinate system

mesh in the computational domain. Furthermore, once a successful finite difference algorithm is developed on the computational domain, any physical domain may be mapped into this region and solved with the same numerical algorithm.

The task of numerical grid generation is to define the transformation between the Cartesian and computational coordinate systems which can be viewed as the development of a curvilinear coordinate system in the physical domain. The general mapping between the two coordinate systems is

$$\xi = \xi\ (x,y) \qquad (2.1.1)$$
$$\eta = \eta\ (x,y)$$

Owing to the discrete nature of finite difference approximations, it is sufficient to define a finite set of coordinate lines, the intersections of which define the grid points in the physical domain. Thus for each grid point (x,y) in the physical domain, there are two curvilinear coordinates (ξ,η) that intersect at that point and designate its location. It is convenient to view this mapping in the computational domain, where for each pair (ξ,η) there corresponds a point (x,y) in the physical domain. This correspondence can be conveniently expressed as the inverse mapping

$$x = x(\xi,\eta) \qquad (2.1.2)$$
$$y = y(\xi,\eta)$$

It is advantageous to designate the curvilinear coordinates with integer values. Finite difference expressions in the computational plane are simplified since $\Delta\xi$ and $\Delta\eta$ become unity and the mapping of eqs. (2.1.2) will appear as a two dimensional array with integer subscripts, creating a naturally ordered way to store the grid points. Using the inverse mapping, the location of any coordinate line in the physical domain can be determined simply as the set of points obtained by holding one of the arguments in the mapping constant. The integer indicies i and j are used here to designated each grid point. Thus the i^{th} point along a ξ coordinate is ξ_i and the j^{th} point along an η coordinate is η_j. For simplicity the notation $(x_{i,j},y_{i,j})$ is used to represent the point $(x(\xi_i,\eta_j),y(\xi_i,\eta_j))$ and the total number of points in each direction ξ and η are IM and JM, respectively. For the projectile problem considered, the ξ and η coordinate lines will always run in the streamwise and normal directions as indicated in Figure 2.1.

Once a mapping is defined, in a discrete sense, the governing equations can be obtained in the computational plane by transforming them onto the curvilinear coordinate system. Derivatives with respect to the Cartesian coordinates can be expressed using eqs. (2.1.1) and the chain rule as

$$\begin{Bmatrix} \frac{\partial}{\partial x} \\ \\ \frac{\partial}{\partial y} \end{Bmatrix} = \begin{bmatrix} \xi_x & \eta_x \\ \\ \xi_y & \eta_y \end{bmatrix} \begin{Bmatrix} \frac{\partial}{\partial \xi} \\ \\ \frac{\partial}{\partial \eta} \end{Bmatrix} \qquad (2.1.3)$$

where ξ_x, ξ_y, η_x, and η_y are the metric coefficients of the transformation. They appear as constants in the transformed equations and depend solely on the mapping of eqs. (2.1.1). Substitution of eqs.

(2.1.3) into the governing equations makes ξ and η the independent spatial variables and now a numerical algorithm can be developed using finite difference approximations on the equally spaced computational domain. Since the metric coefficients depend only on the mapping, virtually any domain may be mapped into the computational domain and solved using the numerical algorithm.

The metric coefficients can be evaluated for a numerically generated transformation by considering the metrics of the inverse transformation. The relationship between the metric and inverse metrics is, for two dimensions,

$$\begin{bmatrix} \xi_x & \eta_y \\ \xi_y & \eta_y \end{bmatrix} = \frac{1}{J} \begin{bmatrix} y_\eta & -y_\xi \\ -x_\eta & x_\xi \end{bmatrix} \tag{2.1.4}$$

where J is the Jacobian of the transformation,

$$J = x_\xi y_\eta - x_\eta y_\xi \tag{2.1.5}$$

The derivation of these equations can be found in reference [13]. Since ξ and η are the independent variables for the inverse metrics, the inverse metrics can be evaluated numerically in the computational domain and subsequently used to determine the metric coefficients.

The primary disadvantage of using boundary fitted curvilinear coordinate systems is the appearance of additional truncation error terms in the finite difference approximations on the computational plane. Mastin [3] has shown, for instance, that the truncation error T for a second order finite difference approximation of the first derivative is,

in the computational domain

$$T = -\frac{1}{6}x_\xi^2 \; f_{xxx} - \frac{1}{2} \; x_{\xi\xi} \; f_{xx} \qquad (2.1.6)$$

The first term is the usual truncation error term containing the grid point spacing x_ξ and the third derivative of the function. The second term is due to the varied spacing along the ξ coordinate lines in the physical domain. It is thus possible to increase the truncation error if the coordinates spacing is not smooth. They have also shown that truncation error for a first derivative varies inversely as the sine of the angle between the curvilinear coordinates and therefore a lack of orthogonality can also increase the truncation error. Orthogonality is also important near solid boundaries for turbulent flow problems because turbulence models often are based on information in the normal direction.

In the context of a coordinate transformation, the adaptive gridding can be viewed as an attempt to reduce the truncation error by adjusting the transformation metrics. This is accomplished by judiciously defining the curvilinear coordinates such that they vary smoothly, are nearly orthogonal, and concentrate in regions where the solution is changing rapidly.

II.2 A Variational Approach

The choice of a method to generate the curvilinear coordinates is arbitrary in that it has no dependence on the partial differential equations that are to be solved. Of the many methods available, however, the variational approach is attractive for purposes of adaptive grid generation because it can provide a simple, explicit means of incorporating desired properties into a grid generation scheme and can

result in a set of elliptic equations which govern the grid. To develop an adaptive grid generation scheme using variational techniques, a measure of each grid characteristic is defined and its integral is taken over the physical domain. If the measures are defined such that they measure a deviation from the desired property then the coordinates ξ and η that minimize the integral will yield a grid with the desired characteristics. As a minimization problem, the Euler-Lagrange equations can be applied to obtain a set of partial differential equations, the solution of which defines the curvilinear coordinates.

The first of three functionals used here is

$$I_p = \int \frac{\nabla\xi \cdot \nabla\xi}{P} dxdy \qquad (2.2.1)$$

which measures the adaptation of the grid spacing in the ξ coordinate direction to the control function P. When P is small, the quantity $\nabla\xi$ must also be small in order to minimize the integral; a small value of $\nabla\xi$ corresponds to large grid spacing. Consequently, when P is large the grid spacing along the ξ coordinate will be small. Similarly, the integral

$$I_q = \int \frac{\nabla\eta \cdot \nabla\eta}{Q} dxdy \qquad (2.2.2)$$

is a measure of the adaptation of the grid spacing in the η direction to the control function Q. The third functional

$$I_o = \int (\nabla\xi \cdot \nabla\eta)^2 \, dxdy \qquad (2.2.3)$$

measures the orthogonality of the ξ and η coordinates. As will be shown later, grid smoothness can be controlled through the definitions of the control functions. These three integrals can be combined into one total functional I_T

$$I_T = \int \left\{ \frac{\nabla\xi \cdot \nabla\xi}{P} + \frac{\nabla\eta \cdot \nabla\eta}{Q} + \lambda(\nabla\xi \cdot \nabla\eta)^2 \right\} dxdy \qquad (2.2.4)$$

where λ is a parameter that weighs the relative importance of adaptation and orthogonality. Before applying the Euler-Lagrange equations for ξ and η, it is beneficial to scale each term in the equations so that they remain of the same order of magnitude. An ordering analysis shows that the first term in eq. (2.2.4) is of order (C^2/P), the second is of order (C^2/Q) and the orthogonality term is of order (C^4/L^2) where C is a computational length scale and L is a physical length scale. It is common practice to use highly stretched grids in transonic flow problems to provide extremely small grid spacing in the viscous sublayer region. The spacing along the normal coordinate can increase by five orders of magnitude, consequently the terms in eq. (2.2.4) may vary in a similar way. It is therefore beneficial to scale the terms locally rather than globally since a global scale cannot reflect the size of the term throughout the domain. The local scales λ_p, λ_q, and λ_o used here are

$$\lambda_p = P_L \qquad \lambda_q = Q_L \qquad \lambda_o = J_L \qquad (2.2.5)$$

where P_L and Q_L are the local values of the control functions and J_L is the local Jacobian and is of order (L^2/C^2). Although they vary throughout the domain they are considered constants during the application of the

Euler-Lagrange equations. This violates the variational principal, but it has been shown [13] that the use of local scaling will produce better grids when there are large changes in the grid spacing. After substituting in the scales, the total functional I_T becomes

$$I_T = \int \{ P_L \frac{\nabla\xi \cdot \nabla\xi}{P} + Q_L \frac{\nabla\eta \cdot \nabla\eta}{Q} + \lambda J_L (\nabla\xi \cdot \nabla\eta)^2 \} dxdy \tag{2.2.6}$$

After applying the Euler-Lagrange equations for ξ and η to the total functional I_T while holding the scales constant, the following set of partial differential equations is obtained:

$$\begin{aligned} &\xi_{xx} + \xi_{yy} + \frac{\nabla P \cdot \nabla\xi}{P} + \lambda(\eta_x^2\xi_{xx} + 2\eta_x\eta_y\xi_{xy} + \eta_y^2\xi_{yy} + \\ &(2\xi_x\eta_x + \xi_y\eta_y)\eta_{xx} + (\eta_x\xi_y + \eta_y\xi_x)\eta_{xy} + (\xi_x\eta_x + 2\xi_y\eta_y)\eta_{yy}) = 0 \\ &\eta_{xx} + \eta_{yy} + \frac{\nabla Q \cdot \nabla\eta}{Q} + \lambda((2\xi_x\eta_x + \xi_y\eta_y)\xi_{xx} + (\eta_x\xi_y + \eta_y\xi_x)\xi_{xy} + \\ &(\xi_x\eta_x + 2\xi_y\eta_y)\xi_{xy} + \xi_x^2\eta_{xx} + 2\xi_x\xi_y\eta_{xy} + \xi_y^2\eta_{yy}) = 0 \end{aligned} \tag{2.2.7}$$

The subscript L has been dropped from the local scales in the differential equations since they are no longer needed to indicate their assumed invariance. The ξ and η coordinates that satisfy these equations will yield a grid with the desired properties defined in the functionals of eqs. (2.2.1), (2.2.2) and (2.2.3). It should be noted that there was no explicit functional to maintain grid smoothness. There are two types of smoothness that should be differentiated. One type, which can be controlled by the control functions, is smoothness of the grid point

spacing along a coordinate line. If the control functions change rapidly along a coordinate, the grid spacing will consequently change rapidly. This smoothing thus can be governed through the definition of the control functions. Another type of smoothness, which is inherent in the elliptic equations, is that between the grid point spacing of adjacent coordinates. If the grid point distribution along two adjacent coordinates varies, the elliptic equations will tend to dampen the variation.

Equations (2.2.7) govern the grid coordinates ξ and η and will be elliptic as long as the parameter λ is not too large. They constitute a boundary value problem and it is therefore necessary to define the coordinates along the boundaries. It is also necessary to adapt the boundary points in a consistent manner with the interior points so that the boundary points remain aligned with the interior points. For this purpose, a one dimensional functional analogous to equation (2.2.1) or (2.2.2) is defined to measure adaptation along a boundary coordinate,

$$I_S = \int \frac{\xi_S^2}{P} \, ds \tag{2.2.8}$$

where s is the arclength along the boundary coordinate. There is no consideration for orthogonality and scaling is unnecessary. Applying the one dimensional Euler-Lagrange equation yields a second order differential equation,

$$\xi_{SS} - \xi_S \, P_S/P = 0 \tag{2.2.9}$$

Note that a similar equation can be obtained for boundary adaptation along a η coordinate by replacing ξ in eq. (2.2.9) with η and P with Q. This one dimensional equation, together with eqs. (2.2.7), are the

equations that govern the grid coordinates in the physical domain.

II.3 Inversion of the Grid Generation Equations

In order to solve the grid generation equations, they are transformed onto the curvilinear coordinate system. This is done by inverting eqs. (2.2.7) and (2.2.9) to make x, y, and s the dependent variables. The curvilinear coordinates then become the independent variables and the grid generation equations are solved on the computational domain which is equivalent to defining the inverse mapping of eqs. (2.1.2). All the advantages of using the computational domain to solve the governing equations discussed previously will be realized here as well.

There are two approaches to inverting the equations. One is to transform the functional I_T onto the computational domain and then apply the Euler-Lagrange equations for x and y. Another approach, used here is to derive the corresponding expressions on the computational domain for each term in eqs. (2.2.7) and (2.2.9) and then to substitute them into the equations. The first term in eq. (2.2.7) can be written using eq. (2.1.3) as

$$\xi_{xx} = \left(\xi_x \frac{\partial}{\partial \xi} + \eta_x \frac{\partial}{\partial \eta}\right) \xi_x \tag{2.3.1}$$

Using eq. (2.1.4) to transform the metrics this expression becomes

$$\xi_{xx} = \left(\frac{y_\eta}{J} \frac{\partial}{\partial \xi} - \frac{y_\xi}{J} \frac{\partial}{\partial \eta}\right) \frac{y_\eta}{J} \tag{2.3.2}$$

and after differentiating, the relationship is found to be

$$\xi_{xx} = \frac{-y_\eta}{J}(y_\eta^2 x_{\xi\xi} - 2y_\xi y_\eta x_{\xi\eta} - y_\xi^2 x_{\eta\eta}) + \frac{x_\eta}{J}(y_\xi^2 y_{\xi\xi} - 2y_\xi y_\eta y_{\xi\eta} + y_\xi^2 y_{\eta\eta}) \tag{2.3.3}$$

Similar expressions can be derived for each of the other second derivative terms in eq. (2.2.7), they are listed in Appendix A. All the first derivative terms, of course, can be transformed using eqs. (2.1.4). By substituting these expressions into eqs. (2.2.7) and collecting like terms the equations for adaptive grid generation become, in the computational domain,

$$\begin{aligned} a_1 x_{\xi\xi} + a_2 x_{\xi\eta} + a_3 x_{\eta\eta} + b_1 y_{\xi\xi} + b_2 y_{\xi\eta} + b_3 y_{\eta\eta} &= S_1 \\ c_1 x_{\xi\xi} + c_2 x_{\xi\eta} + c_3 x_{\eta\eta} + d_1 y_{\xi\xi} + d_2 y_{\xi\eta} + d_3 y_{\eta\eta} &= S_2 \end{aligned} \tag{2.3.4}$$

in which

$$\begin{aligned} a_1 &= -y_\eta(\alpha + \lambda'\beta^2) - 2\lambda' y_\xi \alpha\beta \\ a_2 &= 2y_\eta(\beta + \lambda'\beta\gamma) + \lambda' y_\xi(4\beta^2 + J^2) \\ a_3 &= -y_\eta(\gamma + \lambda'\gamma^2) - 2\lambda' y_\xi \alpha\beta \\ b_1 &= x_\eta(\alpha + \lambda'\beta^2) - 2\lambda' y_\xi \alpha\beta \\ b_2 &= -2x_\eta(\beta + \lambda'\beta\gamma) - \lambda' x_\xi(4\beta^2 + J^2) \\ b_3 &= x_\eta(\gamma + \lambda'\gamma^2) + 2\lambda' y_\xi \gamma\beta \\ c_1 &= y_\xi(\alpha + \lambda'\alpha^2) + 2\lambda' y_\eta \alpha\beta \\ c_2 &= -2y_\xi(\beta + \lambda'\alpha\beta) - \lambda' y_\eta(4\beta^2 + J^2) \\ c_3 &= y_\xi(\gamma + \lambda'\beta^2) + 2\lambda' y_\eta \gamma\beta \\ d_1 &= -x_\xi(\alpha + \lambda'\alpha^2) + 2\lambda' y_\eta \alpha\beta \\ d_2 &= 2x_\xi(\beta^2 + \lambda'\alpha\beta) + \lambda' y_\eta(4\beta^2 + J^2) \\ d_3 &= -y_\xi(\gamma + \lambda'\beta^2) + 2\lambda' y_\eta \gamma\beta \end{aligned} \tag{2.3.5}$$

$$\alpha = x_\eta^2 + y_b^2$$

$$\gamma = x_\xi^2 + y_\xi^2$$

$$\beta = x_\xi y_\xi + x_\eta y_\eta$$

$$\lambda' = \lambda/J$$

and

$$S_1 = \frac{P_\xi J\alpha}{P} + \frac{P_\eta J\beta}{P}$$

$$S_2 = \frac{Q_\xi J\beta}{Q} + \frac{Q_\eta J\gamma}{Q} \qquad (2.3.6)$$

It is important to note that in inverting the equations the derivative of P in the η direction appears even though P was intended to control the spacing in the ξ direction. This occurs because the contravariant base vector $\nabla\xi$ actually indicates the spacing in the direction normal to lines of constant ξ rather than in the ξ direction. If the grid is orthogonal, the two directions, $\nabla\eta$ and $\nabla\xi$, are equivalent and the η derivative will not have any influence since β will vanish. It has been found in numerical experiments that eliminating this term from the equations has a negligible effect and consequently it is dropped from the equations. The source terms S_1 and S_2 in eqs. (2.3.5) then become

$$S_1 = \frac{P_\xi J\alpha}{P}$$

$$S_2 = \frac{Q_\eta J\gamma}{Q} \qquad (2.3.7)$$

The one dimensional equation for the boundary adaptation, eq. (2.2.9), is also transformed in a similar manner. The second derivative term can be shown, using the chain rule, to transform as

$$\xi_{ss} = -\frac{s_{\xi\xi}}{s_\xi{}^3} \tag{2.3.8}$$

Substituting eq. (2.3.7) into eq. (2.2.9) yields the transformed one dimensional equation for adaptation along ξ boundaries

$$s_{\xi\xi} + s_\xi \, P_\xi/P = 0 \tag{2.3.9}$$

and for adaptation along boundaries coincident with η coordinates

$$s_{\eta\eta} + s_\eta Q_\eta/Q = 0 \tag{2.3.10}$$

These equations can be solved directly for the grid point spacing along a coordinate. For instance, integrating eq (2.3.10) once yields the grid point spacing

$$s_\eta = c/Q \tag{2.3.11}$$

where c is a constant of integration and can be shown to be

$$c = \frac{S_T}{\int \frac{d\eta}{Q}} \tag{2.3.12}$$

where S_T is the total arclength along the coordinate in the physical domain. However, since the solution in the interior will require an iterative algorithm, the differential form, eqs. (2.3.9) and (2.3.10) are

retained and solved iteratively so that the boundary points remain consistent with the interior points as the grid evolves.

CHAPTER III
NUMERICAL SOLUTION OF THE ADAPTIVE GRID GENERATION EQUATIONS

III.1 The Newton-Raphson Iterative Method

Equations (2.3.3), (2.3.9) and (2.3.10) constitute a set of nonlinear elliptic partial differential equations. In order to solve these equations numerically they are approximated with second order central difference expressions which results in a set of coupled algebraic equations for each grid point. The finite difference approximation for the equations at grid point, $(x_{i,j}, y_{i,j})$ become

$$a_1(x_{i+1,j} - 2x_{i,j} + x_{i-1,j}) + a_3(x_{i,j+1} - 2x_{i,j} + x_{i,j-1}) + b_1(y_{i+1,j} - 2y_{ij} + y_{i-1,j}) + b_3(y_{i,j+1} - 2y_{i,j} + 2y_{i,j-1}) - F_1 = 0 \tag{3.1.1a}$$

$$c_1(x_{i+1,j} - 2x_{i,j} + x_{i-1,j}) + c_3(x_{i,j+1} - 2x_{i,j} + x_{i,j-1}) + d_1(y_{i+1,j} - 2y_{i,j} + y_{i-1,j}) + d_3(y_{i,j+1} - 2y_{i,j} + y_{i,j-1}) - F_2 = 0 \tag{3.1.1b}$$

$$F_1 = \frac{P_\xi J\alpha}{P} - a_2T_1 - b_2T_2 \tag{3.1.1c}$$

$$F_2 = \frac{Q_\eta J\gamma}{Q} - a_2T_1 - d_2T_2$$

$$T_1 = (x_{i+1,j+1} + x_{i-1,j-1} - x_{i+1,j-1} - x_{i-1,j+1})/4 \tag{3.1.1d}$$

$$T_2 = (y_{i+1,j+1} + y_{i-1,j-1} - y_{i+1,j-1} - y_{i-1,j+1})/4$$

where the source terms and the coefficients all contain first derivatives which when approximated with central differences do not contain the point $x_{i,j}$, $y_{i,j}$ explicitly. This set of equations is solved using a Newton-Raphson iterative scheme. The initial guess for the grid point locations will not satisfy the finite difference representation of the equations and therefore a non-zero residue will exist

$$(R_1)^{\ell}_{i,j} = \text{eq (3.1.1a)} \tag{3.1.2}$$

$$(R_2)^{\ell}_{i,j} = \text{eq (3.1.1b)}$$

where ℓ indicates the iteration number. An estimate of the residue at the next iteration can be obtained by expanding the residue in a Taylor series about the current residue and retaining only the linear terms,

$$(R_1)^{\ell+1}_{i,j} = (R_1)^{\ell}_{i,j} + \Delta x_{i,j} \frac{\partial (R_1)^{\ell}_{i,j}}{\partial x_{i,j}} + \Delta y_{i,j} \frac{\partial (R_1)^{\ell}_{i,j}}{\partial y_{i,j}} \tag{3.1.3}$$

$$(R_2)^{\ell+1}_{i,j} = (R_2)^{\ell}_{i,j} + \Delta x_{i,j} \frac{\partial (R_2)^{\ell}_{i,j}}{\partial x_{i,j}} + \Delta y_{i,j} \frac{\partial (R_2)^{\ell}_{i,j}}{\partial y_{i,j}}$$

where Δx and Δy are the changes in the grid point location

$$\Delta x_{i,j} = x_{i,j}^{\ell+1} - x_{i,j}^{\ell}$$

$$\Delta y_{i,j} = y_{i,j}^{\ell+1} - y_{i,j}^{\ell} \qquad (3.1.4)$$

The derivatives of the residues with respect to the variables $x_{i,j}$ and $y_{i,j}$ can be obtained by differentiating eqs. (3.1.1) and are

$$\frac{\partial (R_1)_{i,j}^{\ell}}{\partial x_{i,j}} = -2(a_1+a_3), \qquad \frac{\partial (R_1)_{i,j}^{\ell}}{\partial y_{i,j}} = -2(b_1+b_3)$$

$$\frac{\partial (R_2)_{i,j}^{\ell}}{\partial x_{i,j}} = -2(c_1+c_3), \qquad \frac{\partial (R_2)_{i,j}^{\ell}}{\partial y_{i,j}} = -2(d_1+d_3) \qquad (3.1.5)$$

An approximate value of $\Delta x_{i,j}$ and $\Delta y_{i,j}$ can be obtained if we require the residue at the next iteration to vanish

$$\begin{Bmatrix} (R_1)_{i,j}^{\ell} \\ \\ (R_2)_{i,j}^{\ell} \end{Bmatrix} = \begin{bmatrix} \frac{\partial (R_1)_{i,j}^{\ell}}{\partial x_{i,j}} & \frac{\partial (R_1)_{i,j}^{\ell}}{\partial y_{i,j}} \\ \\ \frac{\partial (R_2)_{i,j}^{\ell}}{\partial x_{i,j}} & \frac{\partial (R_2)_{i,j}^{\ell}}{\partial y_{i,j}} \end{bmatrix} \begin{Bmatrix} \Delta x_{i,j} \\ \Delta y_{i,j} \end{Bmatrix} \qquad (3.1.6)$$

Substituting eqs. (3.1.5) into eqs. (3.1.6) and solving for $\Delta x_{i,j}$ and $\Delta y_{i,j}$ yields

$$\begin{Bmatrix} \Delta x_{i,j} \\ \Delta y_{i,j} \end{Bmatrix} = \frac{2}{D} \begin{bmatrix} (d_1+d_3) & -(b_1+b_3) \\ -(c_1+c_3) & (a_1+a_3) \end{bmatrix} \begin{Bmatrix} (R_1)_{i,j}^{\ell} \\ (R_2)_{i,j}^{\ell} \end{Bmatrix} \qquad (3.1.7)$$

where

$$D = 4((a_1+a_3)(d_1+d_3)-(c_1+c_3)(d_1+d_3)) \qquad (3.1.8)$$

Knowing Δx_{ij} and Δy_{ij}, the new point location can be calculated by solving eqs. (3.1.4) for $x_{i,j}^{\ell+1}$ and $y_{i,j}^{\ell+1}$.

This same procedure can be applied to the one dimensional equations for adaptation, eqs. (2.3.9) and (2.3.10). The residue for eq. (2.3.9) is

$$(R_S)_i^{\ell} = s_{i+1} - 2s_i + s_{i-1} + s_\xi P_\xi / P \qquad (3.1.9)$$

and the equation for the new point location becomes

$$s_i^{\ell+1} = s_i^{\ell} + \Delta s_i^{\ell} \qquad (3.1.10)$$

$$\Delta s_i^{\ell} = (R_S)_i^{\ell}/2 \qquad (3.1.11)$$

The numerical algorithm consists of updating the value of each point on the boundary with eq. (3.1.11) and of sweeping through the domain and updating each interior point using eq. (3.1.7). The criterion used here to indicate convergence is

$$R_{max} \leq \delta \qquad (3.1.12)$$

where R_{max} is the maximum residue in the interior weighted by the Jacobian

$$R_{max} = \max\left(\frac{\Delta x_{i,j}^2 + \Delta y_{i,j}^2}{J} \right)^{1/2} \qquad (3.1.13)$$

Currently, the criterion δ=0.005 is used.

In the original iterative procedure, the most recently updated values for each grid point location were used in updating the remaining points and thus the procedure resembles a Gauss-Seidel iteration. This approach, however, is not suitable for vector processing and thus cannot benefit from the increased processing speed. Therefore, another approach is used in which all the grid points along a coordinate line are updated in a Jacobi iterative fashion. The new grid point values along that coordinate are then used in updating the points along the next coordinate line. Thus the technique can be viewed as a half Jacobi, half Gauss-Seidel iteration. This technique, which is suitable for vector processing, reduced the computer processing time for each iteration by 40 percent. Furthermore, only a few more iterations were required for the same convergence criterion in comparison to the Gauss-Seidel iteration, thus the effective speed up in processing time is approximately 40 percent.

III.2 Modified Solution Procedure

As pointed out earlier, the grids used for viscous transonic flow problems contain highly refined grid spacing in the direction normal to the surface (η direction) in order to resolve the viscous sublayer. This spacing results in grid cells with aspect ratios, up to 10^5 in the present applications, near solid boundaries. Iterative solution procedures for elliptic equations become inefficient for such large aspect ratios since the motion in one direction will be limited by the small spacing in the other direction. The effect of the aspect ratio on an iterative process can be ascertained by considering the simple grid cell shown in Figure 3.1 which for convenience is rectangular and aligned with the Cartesian

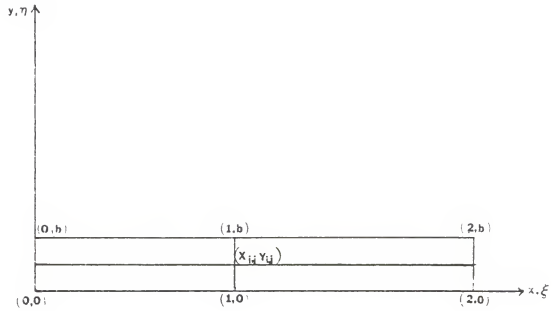


Figure 3.1 High aspect ratio cell

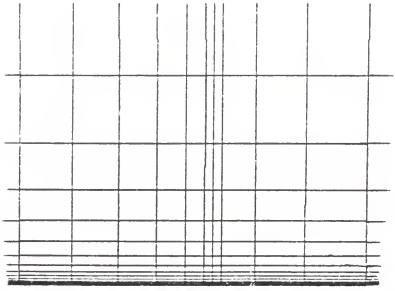


Figure 3.2 Initial grid network

coordinates. For this example the aspect ratio AR is equal to (1/b). Assume for purposes of demonstration that the two terms (Q_η/Q) and (P_ξ/P) are non-zero at the center point and will therefore cause the point to move. We can obtain the changes in the point location by evaluating eqs. (3.1.7) for this simple case. Many terms for this example are zero

$$\begin{aligned} x_\eta=0 \quad y_\xi=0 \quad y_{\xi\eta}=0 \quad y_{\xi\eta}=0 \\ x_{\xi\xi}=0 \quad y_{\xi\xi}=0 \quad x_{\eta\eta}=0 \quad y_{\eta\eta}=0 \end{aligned} \tag{3.2.1}$$

The coefficients become

$$\begin{aligned} a_1=b^3 \quad b_1=0 \quad c_1=0 \quad d_1=b^2 \\ a_2=0 \quad b_2=0 \quad c_2=0 \quad d_2=0 \\ a_3=b \quad b_3=0 \quad c_3=0 \quad d_3=1 \end{aligned} \tag{3.2.2}$$

and upon substitution into eqs. (3.1.7) yield

$$\begin{aligned} \Delta x_{i,j} &= \frac{P_\xi}{2P}\left(\frac{b}{1+b}\right) \\ \Delta y_{i,j} &= \frac{Q_\eta}{2Q}\left(\frac{1}{1+b}\right) \end{aligned} \tag{3.2.3}$$

Writing these in terms of the aspect ratio AR, its effect on the grid point motion becomes clear. If we compare the solution for a square cell , i.e. AR=1,

$$\Delta x_{i,j} = \frac{P_\xi}{2P} \left(\frac{1}{2}\right)$$

$$\Delta y_{i,j} = \frac{Q_\eta}{2Q} \left(\frac{1}{2}\right) \tag{3.2.4}$$

with that for a cell of large aspect ratio AR >> 1,

$$\Delta x_{i,j} = \frac{P_\xi}{2P} \left(\frac{1}{1+AR}\right)$$

$$\Delta y_{i,j} = \frac{Q_\eta}{2Q} \left(\frac{AR}{1+AR}\right) \tag{3.2.5}$$

it becomes evident that the reduction in the grid point motion in the x direction for large AR is proportional to (1/AR). For aspect ratios of the order (10^5) this result becomes prohibitive, rendering the iterative procedure extremely inefficient.

An example has been formulated to demonstrate this deficiency. In an initial grid network, as shown in Figure 3.2, the points along the ξ coordinates are clustered in one region and the points along each η coordinate are clustered near the bottom surface to form the highly stretched spacing typical of grid networks used in transonic flow calculations. The spacing at the surface is approximately 10^{-5} times that in the upper region of the grid. In the adaptive grid generation equations, the control function P is set equal to a constant so that the points will redistribute themselves with equal spacing along the ξ coordinates. Using a technique described in Chapter IV, the control function Q is defined so that the highly stretched spacing along each η

coordinate will be retained in the solution to the adaptive grid generation equations.

Using the Newton-Raphson iterative solution procedure, the solution is advanced 20 iterations which results in the grid shown in Figure 3.3. As can be seen, the point spacing along the ξ coordinates is approximately equal for the coordinates sufficiently above the surface. However, in the region of the high aspect ratio cells, the movement of the points in the ξ direction is retarded by the extremely small spacing in the η direction.

A modification has been made to eq. (3.1.11) which updates the points along the bounding ξ coordinate in formulating this example. The movement of points using the one dimensional equation is independent of the interior points and thus is not restricted by the high aspect ratio cells. This creates an inconsistency at the boundary since the points adjacent to the boundary are constrained by the high aspect ratio cells. Equation (3.1.11) therefore has been modified as

$$\overset{\ell}{\Delta S_i} = \overset{\ell}{(R_{si})_i} \left(\frac{1}{1+AR_i^*}\right) \tag{3.2.6}$$

where AR_i^* is the aspect ratio of the cell adjacent to the i^{th} point along the boundary ξ coordinate. This change only effects the rate of convergence of the iterative solution procedure for the boundary points so that their movement will remain consistent with that of the interior points adjacent to the boundary during the convergence process.

In order to alleviate the deficiency of the above solution procedure, a technique has been developed which is based on solving for a reduced grid in which many of the points in the η coordinate direction have been

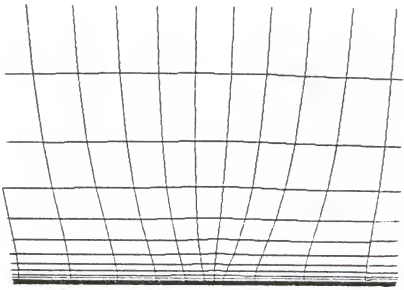


Figure 3.3 Grid network after 20 iterations

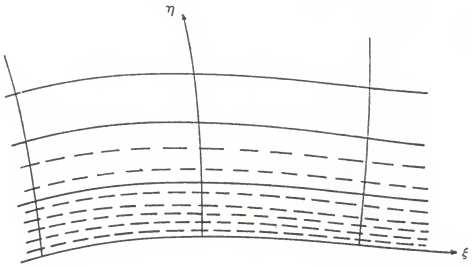


Figure 3.4 Reduced aspect ratio cell

removed to decrease the maximum cell aspect ratio. Figure 3.4 shows a diagram of a typical grid for viscous flow problems containing large aspect ratio cells near a solid boundary. A reduced grid can be formed by removing all the ξ coordinate lines denoted with the dashed lines, resulting in a grid with fewer points along the η coordinate lines which are designated by the intersection of the solid lines. Currently, enough points are removed along the η coordinate lines such that the minimum spacing between ξ coordinate lines is greater than 0.04. The typical spacing along the boundary is of order (0.1) which results in cell aspect ratios of order (1). Upon convergence of the solution for the reduced grid, all the points removed are reinserted along η coordinate lines using the same one dimensional equation that controls the spacing along the boundaries.

It is necessary in this process to adjust the control function Q at the remaining points to account for the spacing requirements of the points removed. A simple procedure for this purpose can be derived by using the equation for one dimensional adaptation, eq. (2.3.10). The procedure will also be adequate for the two dimensional adaptive grid generation equations. If we write explicitly the finite difference approximation for eq (2.3.10),

$$s_{j+1}-2s_j + s_{j-1} + \left(\frac{s_{j+1}-s_{j-1}}{2}\right)\left(\frac{Q_{j+1}-Q_{j-1}}{2}\right)/Q_j=0 \quad (3.2.7)$$

we can regroup the terms in the form

$$(s_{j+1}-s_j)=(s_j-s_{j-1})\left(\frac{Q_j + \left(\frac{Q_{j+1}-Q_{j-1}}{2}\right)}{Q_j - \left(\frac{Q_{j+1}-Q_{j-1}}{2}\right)}\right) \quad (3.2.8)$$

which, after using forward difference expressions to indicate the spacing between two adjacent points can be written

$$\Delta s_j = \Delta s_{j-1}\, C_j \tag{3.2.9}$$

$$C_j = \left(\frac{Q_j + \left(\frac{Q_{j+1}-Q_{j-1}}{2}\right)}{Q_j - \left(\frac{Q_{j+1}-Q_{j-1}}{2}\right)}\right) \tag{3.2.10}$$

Now, as shown in Figure 3.5, if the ith point is removed it will be necessary to adjust the control function such that the new relationship is satisfied

$$\Delta s_{j+1} = \Delta s^*_{j-1}\, C^*_{j-1} \tag{3.2.11}$$

where C^*_{j-1} is the modified control function and ΔS^*_{j-1} is the increased grid point spacing

$$\Delta s^*_{j-1} = \Delta s_{j-1} + \Delta s_j \tag{3.2.12}$$

The correct form of C^*_{j-1} can be obtained in the following derivation

$$\begin{aligned} \Delta s_{j+1} &= \Delta s^*_{j-1}\, C^*_{j-1} \\ \Delta s_j\, C_{j+1} &= (\Delta s_{j-1}+\Delta s_j)\, C^*_{j-1} \\ \Delta s_{j-1}\, C_j\, C_{j+1} &= \Delta s_{j-1}\, (1+C_j)\, C^*_{j-1} \end{aligned} \tag{3.2.13}$$

$$\frac{C_j C_{j+1}}{1+C_j} = C^*_{j-1}$$

Once C^*_{j-1} is calculated using eq. (3.2.13), the modified value of Q_{j-1} can be obtained by solving eq. (3.2.10) for Q^*_{j-1}

$$Q^*_{j-1} = \left(\frac{1+C^*_{j-1}}{1-C^*_{j-1}}\right)(Q_{j+1}-Q_{j-2}) \qquad (3.2.14)$$

This procedure is applied for each point along an η coordinate that is removed. It should be noted that this procedure does not affect the final grid, the points remaining in the reduced grid will obtain the same location as if all the points were used in the iterative solution procedure. The removed points are, once the solution for the reduced grid is obtained, inserted back along the η coordinates lines. This is done by approximating the η coordinate with straight line segments between the points of the reduced grid and then reinserting the removed points along each segment with spacing dictated by eqs. (2.3.10) using the original control function Q. If necessary, the use of a higher order approximation to the η coordinate lines, such as a cubic spline, can be used to obtain a better representation of the η coordinate lines. In using this procedure with the two dimensional grid generation equations, the solution may vary somewhat since the equations are coupled. Also, the effects of orthogonality may alter the results in some areas.

In order to show the advantage of this procedure, the initial grid network of the previous example is advanced 20 iterations using the reduced grid technique. Figure 3.6 shows the initial reduced grid in which the points responsible for the high aspect ratio cells are removed. After modifying the control function Q, the solution to the adaptive grid generation equations is advanced 20 iterations resulting in the grid shown in Figure 3.7. As can be seen, the points along all ξ coordinates

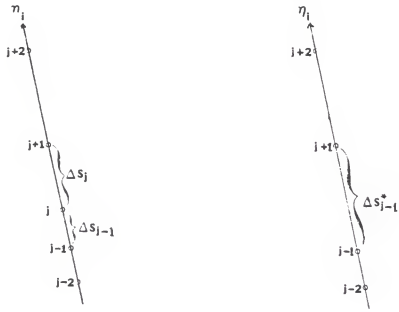


Figure 3.5 Corresponding grid point spacing

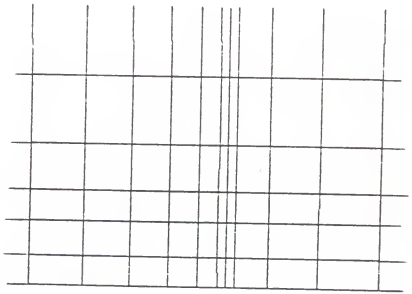


Figure 3.6 Initial reduced grid network

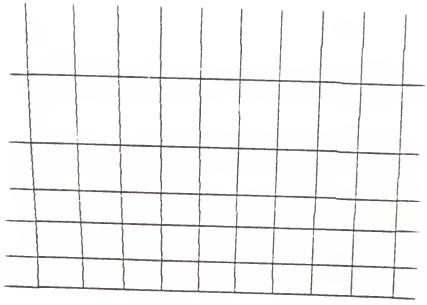


Figure 3.7 Reduced grid network after 20 iterations

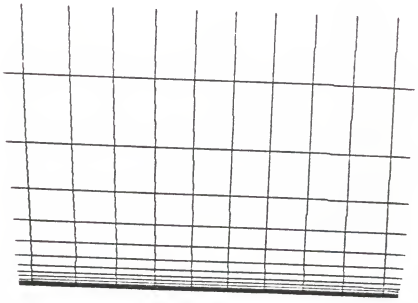


Figure 3.8 Final grid network after reinserting grid points

are approximately equally spaced and the spacing along the η coordinate lines in the initial reduced grid is retained which indicates that the modified control function is correct. Upon inserting the removed points along the η coordinates using eq. (2.3.12), the final grid network, Figure 3.8, is obtained, which is the same grid that would have been obtained in Figure 3.3 after a large number of iterations.

III.3 The Adaptive Grid Generation Code

A computer code base on the iterative solution procedure described in the preceding section has been developed and is referred to here as the Adaptive Grid code. As the solution process is iterative, an initial guess for the grid point locations is required and the flow variables for which the grid network is to be adapted are needed for calculation of the control functions. These are considered as input to the Adaptive Grid code. The initial guess for the grid network can be a grid network obtained by any conventional grid generation technique or a previously adapted grid network. Values for the flow variables at the grid points are obtained using the flow solvers which are described in Chapter V. The sequence of calculations in the Adaptive Grid code proceeds as follows:

1. Input an initial grid network and the flow variables at each grid point.
2. Calculate the control functions P and Q.
3. Remove points to form the reduced grid network and calculate the modified control function Q^*.
4. Solve the grid generation equations for the reduced grid network using P and Q^*.
5. Reinsert the removed points along the η coordinate lines using the control function Q which then results in the adapted grid network.

The methods used to calculate the control functions from the solution variables is described in Chapter IV.

CHAPTER IV
CONTROL FUNCTIONS

IV.1 The Basic Form

It is the role of the control functions in the equations governing grid adaptation to dictate the need for increased resolution and, as indicated in section II.1, they should be judicially chosen so that the resulting grid resolution reduces what would otherwise be large truncation error. For instance, Pierson and Kutler [14] chose for the control function, when solving a one dimensional inviscid Burger's equation, the third derivative of the dependent variable which is also the leading truncation error term in their finite difference approximation to Burger's equation. Their one dimensional adaptive grid scheme was thus an attempt to minimize a measure of the truncation error over the domain. In more complex problems, however, with more dependent variables and in higher dimensions, the truncation error expressions are complex and cumbersome to calculate. Therefore, in practice, an understanding of the physical problem and experience guide the choices for the control functions. The general control function considered here is

$$\begin{aligned} P &= 1 + \gamma_p f \\ Q &= 1 + \gamma_q g \end{aligned} \tag{4.1.1}$$

where γ_p and γ_q are parameters and f and g are some derivatives of the dependent variables scaled to range between zero and one. In the following developments, the terms W,γ and h will be used to indicate either P,γ_p and f or Q,γ_q and g. By evaluating numerically the solution to the one dimensional equation for adaptation, the following expression equating the control function to the grid point spacing between points (i) and (i+1) along a coordinate line is obtained,

$$\Delta s_i = \frac{S_T \frac{1}{1+\gamma h_i}}{\sum_j \frac{1}{1+\gamma h_j}} \tag{4.1.2}$$

After evaluating this expression for the maximum and minimum spacing corresponding to h = 1 and h = 0 respectively, and forming their ratio

$$\frac{(\Delta s_i)_{max}}{(\Delta s_i)_{mim}} = 1 + \gamma \tag{4.1.3}$$

it can be seen that the specification of γ dictates the ratio of maximum and minimum spacing along a coordinate line. The parameter γ also influences the smoothness of the resulting grid point distribution. If it is chosen as zero, equal spacing as indicated by eq. (4.1.2) would result; as γ is increased, the spacing must change more rapidly to obtain the varied spacing. Following the work of Nakahashi and Deiwert [7], another parameter σ has been introduced to obtain more control over the spacing. The general control function now becomes

$$W = 1 + \gamma h^{\sigma} \tag{4.1.4}$$

By prescribing the minimum spacing and γ, eq (4.1.2) can be written as

$$(\Delta s_i)_{min} \sum_j \left(\frac{1}{1+\gamma h_j^{\sigma}}\right) - \frac{S_T}{1+\gamma} = 0 \qquad (4.1.5)$$

in which σ is the unknown quantity and can be determined by a root finding technique. Currently, the Newton-Raphson finding method is employed. Thus, both the minimum and maximum grid point spacing along a coordinate can be specified by specifying the two quantities, $(\Delta s_i)_{min}$ and γ. It should be noted that the prescribed values may not be realized in practice due to the coupling effects of the two dimensional adaptive grid generation equations. Also, the constraint of orthogonality may alter the results in some areas.

IV.2 Smoothing

The choices for the functions f and g will of course depend on the problem being solved. However, the functions will in general, contain derivatives of the solution which are approximated numerically and it is therefore useful to smooth the control functions to eliminate any spurious data. Also, since smoothing is most effective when the function changes most rapidly, it will help reduce any rapid variations in the control function and subsequently smooth the grid point spacing. The method of smoothing used here, which is similar to that used in reference [5] is

$$W_{i,j} = W_{i,j} + \omega\left(\frac{W_{i+1,j}+W_{i-1,j}}{2} - W_{i,j}\right), \quad \begin{matrix} i=2,IM-1 \\ j=1,JM \end{matrix}$$

(4.2.1)

$$W_{i,j} = W_{i,j} + \omega\left(\frac{W_{i,j+1}+W_{i-1,j}}{2} - W_{i,j}\right), \quad \begin{matrix} i=1,IM \\ j=2,JM-1 \end{matrix}$$

where the value of ω used here is 0.4. The control function at a point is replaced by a weighted average of itself and the two adjacent points along a coordinate line. Currently, two consecutive sweeps of this algorithm are made over the domain for each of the control functions, P and Q. In this algorithm there is no accounting for the varying spacing between points and thus this smoothing is in the computational plane.

IV.3 Other Control Functions

In many instances a certain point distribution in one coordinate direction is known to yield good results and it would be advantageous to maintain this prescribed distribution in that direction while allowing adaptation to the solution in the other directions. For instance, in some of the viscous transonic flow problems solved here, it is known that a distribution in the direction normal to the surface based on an exponential function will yield good results provided enough points are used. A procedure to incorporate specified point distributions into the adaptive grid scheme is easily developed by using eq. (2.3.10).

Consider some given grid point distribution s in the η coordinate direction which is to be reproduced by the adaptive grid generation equations

$$s = s(\eta) \tag{4.3.1}$$

After solving for the control function, eq. (2.3.10) becomes

$$Q_i = \frac{1}{(s_\eta)_j} \tag{4.3.2}$$

The control function then can be determined from the prescribed distribution s by using central difference approximations to evaluate S_η. As noted previously, the resulting distribution may vary somewhat due to the coupling of the two dimensional adaptive grid equations and effects of orthogonality. This approach, which is used in some of the cases solved here to specify the spacing in the normal coordinate direction, is similar to the technique of Thompson et al. [15] to obtain boundary point distribution in the domain interior. The equation for the grid point spacing used to obtain the arclength distribution is

$$\Delta_{sj} = \Delta s^*(1+\epsilon)^n \tag{4.3.3}$$

where ϵ is determined so that S_T matches the total length of the coordinate line (see reference 16), Δs^* is a specified spacing at the projectile surface, and n is equal to the index j.

The actual choices for the functions h and g in eqs. (4.1.4) are discussed in Chapter VIII.

CHAPTER V
FLOW SOLVERS

V.1 The Governing Equations

The governing equations for viscous transonic flow are the Navier-Stokes equations and an equation of state. The Navier Stokes equations, which represent the conservation of mass, momentum and energy form a set of parabolic/hyperbolic partial differential equations when the time dependent terms are retained. To solve these equations numerically on an arbitrary domain the governing equations are scaled and then transformed onto the curvilinear coordinates. For problems in three dimensions, three curvilinear coordinates are required, denoted here as ξ, η and ζ. However, for bodies of revolution, such as the axisymmetric projectiles at zero angle of attack considered here, the solution is assumed invariant in the circumferential direction, designated as the ζ direction, and the equations can be reduced since only two independent spatial variables are required. A further reduction in the equations comes from the thin layer approximation. In the thin layer approximation all the viscous derivatives in the direction along a solid surface (the ξ direction in the present applications) are dropped from the equations. Due to restrictions on computer storage and CPU time, it is not usually feasible to resolve the viscous terms in both the directions normal to the surface and along the surface. For high Reynolds number flows, such

as the transonic flow considered here, large velocity gradients are expected to occur normal to the surface and most of the grid points are used to resolve these gradients. As larger spacing is then used along the surface and the viscous derivatives in that direction are expected to be small in comparison, they are dropped from the equations.

The conservative form of the transformed thin layer governing equations for axisymmetric flow, written in the compact vector form are

$$\partial_t \hat{Q} + \partial_\xi \hat{E} + \partial_\eta \hat{F} + \hat{G} = R_e^{-1} \partial_\eta \hat{S} \qquad (5.1.1)$$

where

$$\hat{Q}=J^* \begin{Bmatrix} \rho \\ \rho u \\ \rho v \\ \rho w \\ c \end{Bmatrix}, \quad \hat{E}=J^* \begin{Bmatrix} \rho U \\ \rho uU+\xi_x p \\ \rho vU+\xi_y p \\ \rho wU+\xi_z p \\ (e+p)U \end{Bmatrix}, \quad \hat{F}=J^* \begin{Bmatrix} \rho V \\ \rho uV+\eta_x p \\ \rho vV+\eta_y p \\ \rho wV+\eta_z p \\ (e+p)V \end{Bmatrix},$$

$$\hat{G}=J^* \begin{Bmatrix} 0 \\ 0 \\ -\rho w^2 y - \rho/y \\ \rho w(y_\xi U+y_\eta V) \\ 0 \end{Bmatrix}, \qquad (5.1.2)$$

$$\hat{S} = J^* \begin{Bmatrix} 0 \\ uRu_\eta + (u/3)T^\eta x \\ uRv_\eta + (v/3)T^\eta y \\ uRw_\eta + (w/3)T^\eta z \\ A(\tfrac{1}{2}\mu(u^2+v^2+w^2)_\eta + uP_r^{-1}(\gamma-1)^{-1}(a^2)_\eta \\ \quad +(1/3)(\eta_x u+\eta_y v+\eta_z w)T \end{Bmatrix},$$

and

$$R = \eta_x^2 + \eta_y^2 + \eta_z^2$$

$$T = \eta_x u_\eta + \eta_y v_\eta + \eta_x w_\eta$$

Here, ρ is the density, p is the pressure and E is the total specific energy. The velocities u,v and w correspond to the x, y and z directions, respectively, as indicated in Figure 5.1 and U, V, and W are the contravariant velocity components corresponding to the ξ, η and ζ directions respectively. The coefficient of viscosity is μ, Pr is the Prandtl number, Re is the Reynolds number, γ is the ratio of specific heats and a is the local speed of sound. The details of the transformation and application of the thin layer approximation to obtain eqs. (5.1.1) are available in reference [13]. The assumption of a calorically and thermally perfect gas was made in closing the system of equations. The coefficient of viscosity μ is considered variable and it is also necessary to account for the effects of turbulence for the present applications. Under the assumption of ζ invariance, only a two dimensional grid network containing ξ and η coordinates in a plane through the axis of symmetry will be required. It should be noted, however, that the metric coefficients represent the three dimensional coordinate transformation and are therefore different from the expressions of eqs. (2.1.4) and the Jacobian J^* now represents the grid cell volume.

V.2 Solution Algorithms for the Governing Equations

The implicit spatially factored numerical scheme of Beam and Warming [17] is used to solve the transformed governing equations. The scheme is implemented for three dimensional problems in a code due to Steger and Pulliam [18]; the code used here is a modified version which was developed by Nietubicz et al. [19] for the efficient calculation of axisymmetric projectile problems and is referred to here as the Thin

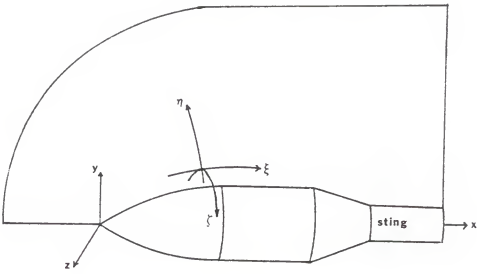


Figure 5.1 Coordinate systems for projectile configuration

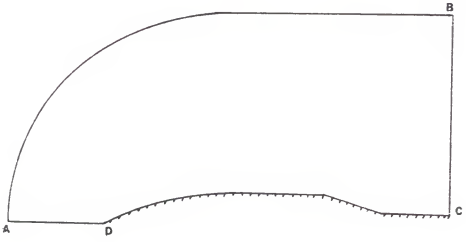


Figure 5.2 Boundary configuration for projectile with sting

Layer code. The code employs Sutherland's law [20] to relate the viscosity and heat conductivity to the temperature and the effects of turbulence are modeled using the eddy viscosity model of Baldwin and Lomax [21].

The numerical algorithm is a time-marching scheme in which steady state solutions are obtained as the time asymptotic solution of the unsteady equations. Spatial derivatives are approximated with central differences and a forward difference approximation is used for the unsteady terms. The time marching scheme of the finite difference approximation to the governing equations are written in delta form as

$$(I + h\,\delta_\xi \hat{A} - \epsilon_I D_1)\,(I + h\delta_\eta \hat{B} - hR\bar{e}^1 J\hat{M}J^1 - \epsilon_I D_2)\Delta\hat{Q}$$
$$= -\,h(d_\xi \hat{E} + \delta_\eta \hat{F} - R\bar{e}^1\,\delta_\eta \hat{S} - \hat{G}) - \epsilon_E D_3 \qquad (5.2.1)$$

where A and B are the Jacobian matrices

$$\hat{A} = \frac{\partial \hat{E}}{\partial \hat{Q}} \qquad \hat{B} = \frac{\partial \hat{F}}{\partial \hat{Q}} \qquad (5.2.2)$$

and M is obtained as the time linearization of S. Here, δ is the central difference operator. Details of this scheme can be found in Warming and Beam's [17] work, as well as in that of others. The terms D_1, and D_2 are implicit dissipation operators and D_3 is an explicit dissipation term

$$D_1 = J^{-1}\,\Delta_\xi\,\nabla_\xi\,J$$
$$D_2 = J^{-1}\,\Delta_\eta\,\nabla_\eta\,J \qquad (5.2.3)$$
$$D_3 = J^{-1}\,((\Delta_\xi\,\nabla_\xi)^2 + \Delta_\eta\,\nabla_\eta)^2)\,J\,\hat{Q}$$

where D and ∇ are forward and backward difference operators, respectively. Since the scheme is not naturally dissipative these terms are added to the finite difference scheme to prevent nonlinear instabilities and high frequency spatial oscillations. The constant coefficients ϵ_I and ϵ_E used here are $4\Delta t$ and $2\Delta t$ respectively which are consistent with values used in other applications of this code [19]. In general, these coefficients should be chosen large enough to dampen the instabilities but kept small enough to prevent any degradation of solution accuracy.

In Chapter VII, some examples using the Thin Layer code with the Adaptive Grid code show that there are problems in obtaining a converged solution for the thin layer equations when the grid is adapted. Therefore another numerical scheme for solving the thin layer equations is also used. This scheme is the TVD scheme developed by Yee [22] for the multidimensional Navier-Stokes equations. The theory and development of TVD schemes is beyond the scope of current discussions. However, for purposes here it is sufficient to view the TVD scheme as an improvement in the artificial damping terms in the Beam and Warming scheme. In fact, the TVD scheme developed by Yee for multidimensions can be implemented into the existing Thin Layer code based on the Beam and Warming scheme by modifying only the added dissipation terms. The original Thin-Layer code has been modified to provide an option for the TVD scheme [23] and is referred to here as the TVD code. These 'sophisticated' dissipation terms switch the scheme from second order to first order accuracy at points of extrema and result in a more dissipative scheme in those regions. It should be pointed out that the primary purpose of the current research is to develop an adaptive grid generation technique. The TVD scheme is used

here because of problems occurring when using the Beam and Warming scheme in conjunction with the adaptive grid generation technique. These problems are discussed more fully in Chapter VII.

The first flow problem considered is a the axisymmetric projectile flow problem with an attached sting to eliminate the base flow region. In both the Thin Layer code and the TVD code the boundary conditions are implemented explicitly by updating the flow variables at the boundary after each time step. Referring to Figure 5.2 which shows the basic C-type grid configuration for the axisymmetric projectile flow problem, there are four different sets of conditions applied at the four bounding coordinates. Along the outer boundary, line AB, free stream conditions are assumed. Thus the outer boundary is required to be sufficiently far from the projectile so that this assumption is valid. Along the upstream line of symmetry, line AD, the variables are obtained by requiring symmetry about the x axis. Numerically this is done by setting a second order forward difference approximation to the first derivative along the ξ coordinate for each variable equal to zero and solving for the values at the points on the axis in terms of known values in the domain. Along the downstream boundary, line BC, the variables are obtained by extrapolation along the ξ coordinate lines. However, if the free stream velocity is subsonic the pressure is fixed. This is done to prevent oscillations in the pressure distribution that would otherwise occur [24]. Along the projectile surface, line CD, the no slip condition holds for viscous flow, thus the velocity components are zero. The density is obtained by zero order extrapolation along the η coordinate lines and the pressure is obtained by satisfying the momentum equation along the surface.

Another problem considered is the projectile base flow problem. In Figure 5.3, which shows the basic O-type grid configuration, the ξ coordinate lines bend around the sharp corner at the base and end on a downstream axis of symmetry which is now an η coordinate. The existing codes are used for this problem by simply modifying some of the boundary conditions. The boundary conditions along the upstream axis of symmetry, line AD, and the projectile surface, line CD, are the same as those used in the projectile problem with sting. For the downstream axis of symmetry, line BC, the same conditions along the upstream axis are used; however, a backward finite difference formula is required. The freestream conditions are applied along the outer boundary along the line AA' and the downstream boundary conditions for the projectile with sting problem are applied along the portion of the outer boundary between A'B, except that the extrapolation is done along the η coordinate lines.

V.3 Examples on Fixed Grid Networks

In order to show the general characteristics of the Beam and Warming and TVD schemes, and for purposes of comparison with the solutions obtained with the adaptive grid generation technique, the two schemes were used to solve the projectile flow problem with sting on a fixed grid network. The fixed grid network was obtained using a method developed by Steger et al. [16] which is an effective technique for external flow problems. Two equations, one enforcing orthogonality and one specifying the grid cell volume, form a set of hyperbolic partial differential equations which are solved numerically to generate the grid network. In the marching solution algorithm, the initial grid points are specified along the projectile surface. The solution is marched outward in the η

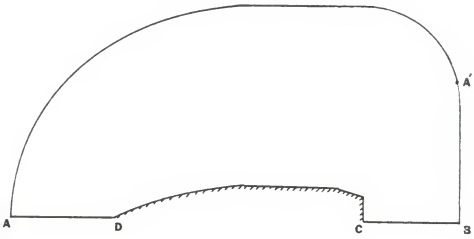


Figure 5.3 Boundary configuration for projectile base flow problem

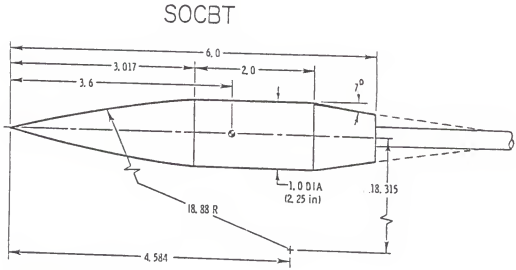


Figure 5.4 SOCBT projectile configuration

coordinate direction, defining each successive ξ coordinate line such that the spacing between each ξ coordinate line is consistent with the prescribed grid cell volume. At the bounding η coordinate lines, the upstream axis of symmetry and the downstream boundary of Figure 5.2, the ξ coordinate lines are required to intersect the η coordinates at right angles.

The projectile used in all the calculations is a 6-caliber, secant-ogive cylinder boattail configuration which is shown in Figure 5.4. There are experimentally determined pressure coefficients available at a series of points along the projectile surface for a range of Mach numbers in the transonic range [25]. The data will serve as a means of comparing the computed solutions. A grid network for this projectile problem with sting was generated using this technique with 70 points in the ξ coordinate direction (IM=70) and 60 points in the η coordinate direction (JM=60). As shown in Figure 5.5, most of the points in the η coordinate direction were clustered near the projectile surface to resolve the boundary layer. The spacing at the projectile surface in the η coordinate direction is 0.00002. The outer boundary is extended to four times the projectile length. In the streamwise direction, the points were clustered near the secant-ogive/cylinder and cylinder/boattail junctures in anticipation of the expansion waves that will occur in transonic flow.

The first solution on this grid network is obtained with the Thin Layer code for Mach 0.96. The Reynolds number for this and all examples is 76,000. For all the results obtained, the time step used for a converged solution is 0.1. Figures 5.6 and 5.7 show the convergence process of the solution algorithm for the indicated time spans. It is quite obvious that the solution oscillates in the region containing the

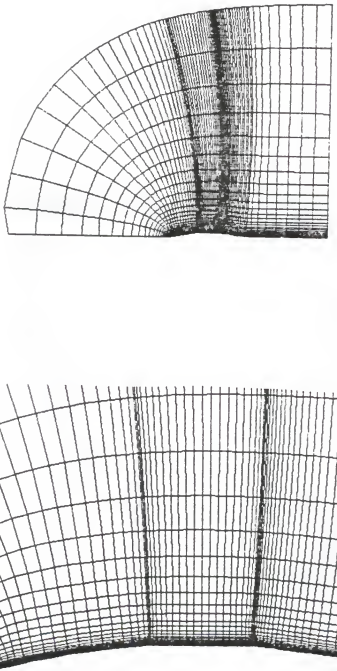


Figure 5.5 Initial fixed grid network for projectile with sting (70 x 60)

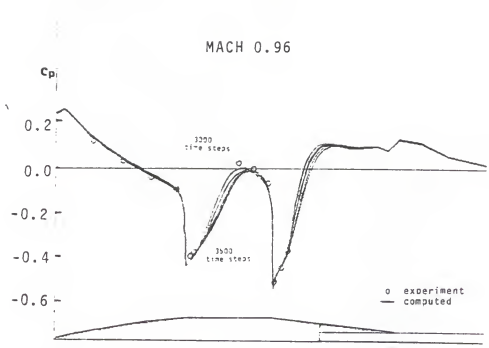


Figure 5.6 Converging solution for the Beam and Warming scheme

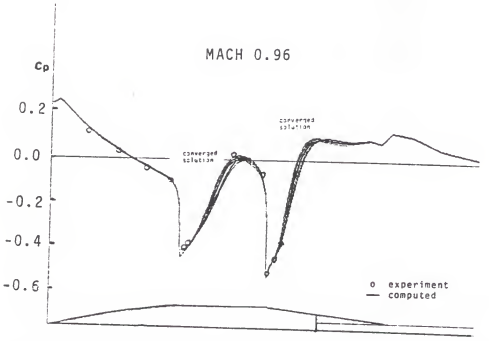


Figure 5.7 Converged solution for the Beam and Warming scheme

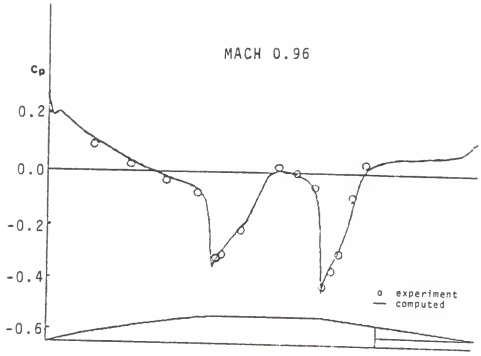


Figure 5.8 Converged solution for the TVD scheme

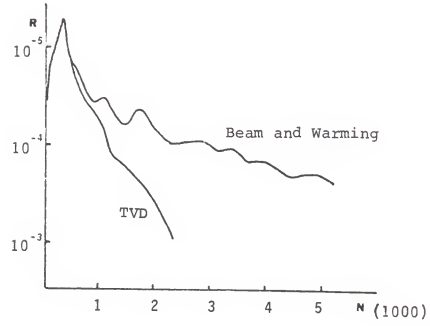
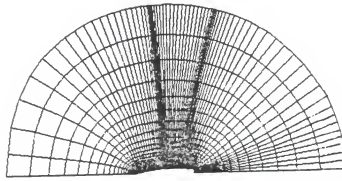


Figure 5.9 Least squared residue

shocks. However, after approximately 5000 time steps the solution does converge and agrees well with the experimental data as indicated by the comparison of the experimental and computed pressure coefficient along the projectile surface. Figure 5.8 shows the results obtained using the TVD code. The solution also agrees well with the experimental data; moreover it converged within 1600 time steps. This difference in the convergence ofthe two schemes is shown in Figure 5.9 which plots the least square residue R, versus the number of time steps N. Here the slower, oscillatory convergence of the Beam and Warming scheme is evident while the TVD scheme appears to converge monotonically.

The projectile base flow problem is also solved on a fixed grid using the TVD code. The grid network for this calculation was obtained using a modified version of the technique of Steger et al. [16] in which the boundary condition on the downstream bounding η coordinate line has been changed so that the ξ coordinate lines intersect with the x axis at right angles. The grid network obtained with IM=70 and JM=60 is shown in Figure 5.10. As can be seen, the ξ coordinate lines now bend around the sharp corner, and end at the downstream axis of symmetry forming an O-type grid network. The computed pressure coefficient obtained for this problem using the TVD code is shown in Figure 5.11. Again, the solution converged in approximately 1600 time steps. It compares well, as before, over the secant-ogive and cylinder portions but then differs from the experimental results at the end of the boattail. No experimental data is available for the base region. There are a number of reasons that may cause the difference between the computed and experimental results near the corner. One reason may be the presence of a narrow sting used to support the experimental model that is not present in the computational

configuration. The difference could also be due to poor resolution in the projectile corner region or an inadequate turbulence model near the flow separation region.



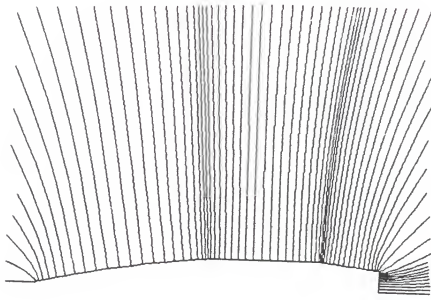


Figure 5.10 Initial fixed grid network for projectile base flow problem (70 x 60)

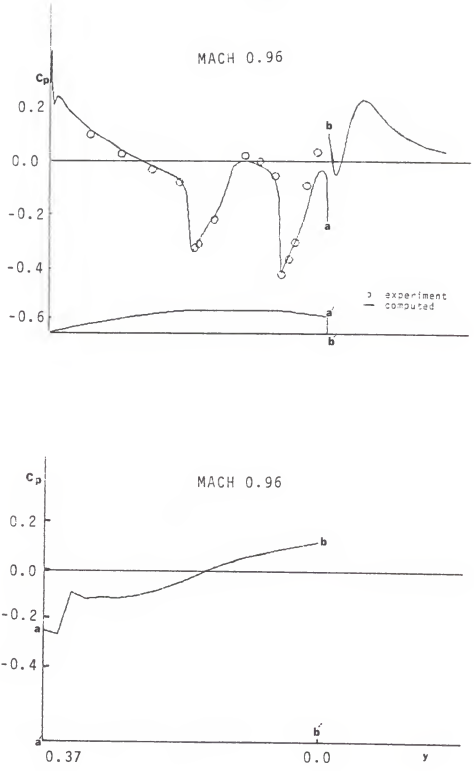


Figure 5.11 Converged solution for projectile base flow problem using the TVD scheme

CHAPTER VI
PRELIMINARY TESTS AND MODIFICATIONS

VI.1 The General Procedure

Prior to coupling the Adaptive Grid code with the flow solvers to solve the projectile flow problem, it was found beneficial to modify the adaptive grid generation method. The following sections include an example showing the control over orthogonality and then describes two modifications made to the adaptive grid scheme to improve the general technique. It should be noted that each of the modifications is made to improve the method for the present flow problem and geometry and may not be necessary or useful in other problems.

For each example the initial grid network used was that of Figure 5.5, or that of 5.10 if the base flow configuration is used in the example. In the adaptive grid code, the control function P was set equal to a constant so that the clustered points along each ξ coordinate line in the initial grid would tend to spread and equal spacing would result. The control function Q was defined using eqs. (4.3.2) with the distribution of eq. (4.3.3) so that the same point distribution along the η coordinates in the initial grid would be retained in the grid obtained using the Adaptive Grid code. Each of the figures used in the examples show only the points in the reduced grid network.

VI.2 Orthogonality

In order to demonstrate the effect of enforcing orthogonality, two cases were considered, one with the parameter $\lambda = 0.0$, the second with $\lambda = 0.5$. For the present projectile, the most difficult area to obtain an orthogonal grid is over the projectile nose. This is due to the sharp nose configuration which results in the two boundaries, the projectile surface and the upstream axis of symmetry, meeting at an obtuse angle. The two sets of coordinates, which run parallel to the boundaries, then tend to intersect at similar angles. This result is shown in Figure 6.1a, which shows an enlarged view of the grid in the nose section that resulted when $\lambda = 0.0$. To improve the orthogonality in this region, another case for $\lambda = 0.5$ was run, and as the results of Figure 6.1b indicate, the orthogonality of the grid coordinates near the nose was in general increased. For all the solutions to the transonic flow problem using the adaptive grid technique reported in Chapter VIII, λ was set equal to 0.5.

VI.3 Curvature

One inherent result of grids obtained as solutions to equations based on the Laplacian operators is the effect of curved boundaries on the spacing of interior coordinate lines. In general, boundaries that are convex will attract coordinate lines and those that are concave will repel coordinate lines. In the context of the adaptive grid generation scheme presented here, this effect will alter the grid spacing dictated by the control functions, either increasing or decreasing the spacing that would result in the absence of curvature effects. The severity of the boundary curvature's influence on the grid is best seen in solving

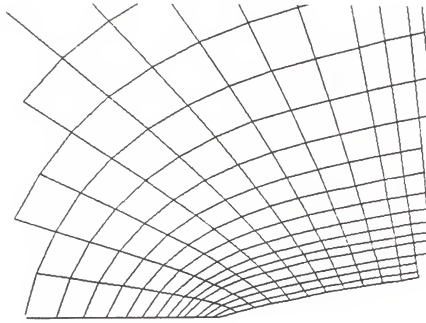


Figure 6.1a Grid network using $\lambda = 0.0$

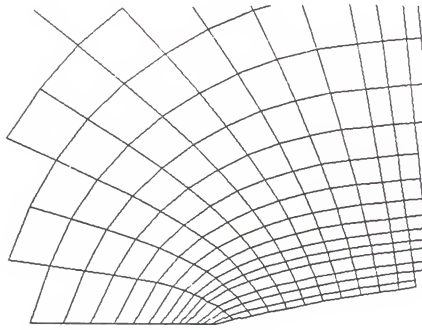


Figure 6.1b Grid network using $\lambda = 0.5$

the adaptive grid generation equations for the base flow grid network. Recall that thel control function Q is defined so that the original grid point spacing in the normal direction would be retained. However, the point spacing resulting as a solution to the adaptive grid equations differs along each η coordinate line as shown in Figure 6.2a. The attraction of points is most prominent near the sharp corner at the projectile base where the effect is large due to the high curvature at the corner.

A simple and effective technique to eliminate the effect of curvature can be developed by considering the solution of Laplace's equations between two concentric circles C_1 and C_2 with radii R_1 and R_2 respectively.

$$\begin{aligned} \nabla^2 \xi &= 0 \\ \nabla^2 \eta &= 0 \end{aligned} \qquad (6.2.1)$$

Solving Laplace's equation is equivalent to solving the adaptive grid equations with constant control functions and neglecting orthogonality (i.e. $\lambda = 0.0$). In using constant control functions, we expect the coordinates to be equally spaced in each direction. It is advantageous to write the equations in polar coordinates (r,θ), and it is sufficient here to seek a solution in which q is a function of θ only and η is a function of r only. The ξ coordinate lines then align themselves with the θ coordinate direction and the η coordinate lines become aligned with the radial coordinate r. The problem can then be formulated as

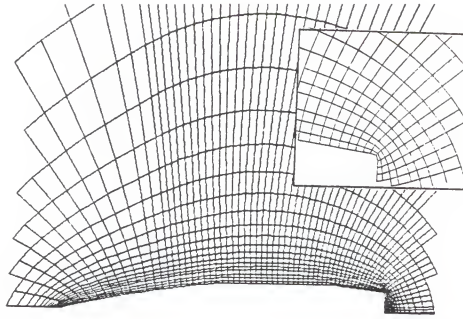


Figure 6.2a Grid network obtained without corrections for curvature

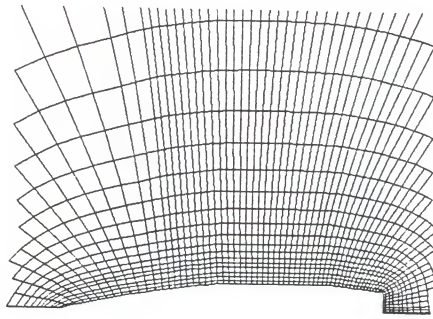


Figure 6.2b Grid network obtained with corrections for curvature

$$\frac{d^2\xi}{d\theta^2} = 0$$

$$\frac{d^2\eta}{dr^2} + \frac{1}{r}\frac{d\eta}{dr} = 0 \qquad (6.2.2)$$

$$\xi = \theta \text{ on } C_1 \text{ and } C_2$$

$$\eta = R_1 \text{ on } C_1$$

$$\eta = R_2 \text{ on } C_2$$

The boundary conditions corresponds to uniform spacing along the ξ coordinate lines coincident with the bounding concentric circles. The spacing along each coordinate direction is the inverse of the first derivatives ξ_θ and η_r which by integrating eqs. (6.2.2) are found to be

$$\xi_\theta = a$$
$$\eta_r = \frac{b}{r} \qquad (6.2.3)$$

where a and b are constants of integration. These results indicate that the spacing in the ξ direction is uniform while the spacing in the η direction is smaller near the inner circle C_1 and larger near the outer circle C_2 . The term in the governing equation responsible for this result, designated here as K, is

$$K = \frac{1}{r}\frac{d\eta}{dr} \qquad (6.2.4)$$

which can be viewed as the curvature of the intersecting ξ coordinate line, (1/r), multiplied by the inverse of the spacing along the η

coordinate line. The effects of the curved ξ coordinates on the spacing along the η coordinate lines can be eliminated for this problem simply by subtracting K from the governing equation for η,

$$\nabla^2 \xi = 0$$

$$\nabla^2 \eta - \frac{1}{r} \frac{d\eta}{dr} = 0 \qquad (6.2.5)$$

This technique can be extended to the general adaptive grid equations by developing the curvature term K for each curvilinear coordinate line in the computational domain and then subtracting them from eqs. (2.2.7). The term needed to cancel the effects of a curved η coordinate line on the spacing along a ξ coordinate line is comprised of the curvature of the η coordinate line and the spacing along the ξ coordinate line. The curvature ρ of a η coordinate line is defined as

$$\rho = \frac{d}{ds} \left| \bar{\tau} \right| \qquad (6.2.6)$$

where $\bar{\tau}$ is a unit tangent to the η coordinate and is in the computational plane

$$\bar{\tau} = \frac{(x_\eta, y_\eta)}{\sqrt{\alpha}} \qquad (6.2.7)$$

where α is defined in eq. (2.3.5). By using the two relationships

$$S_\eta = \sqrt{\alpha} \qquad \frac{d}{ds} = \frac{1}{s_\eta} \frac{d}{d\eta} \qquad (6.2.8)$$

the curvature ρ can be shown to be

$$\rho = \left(\frac{y_\eta x_{\eta\eta} - x_\eta y_{\eta\eta}}{\alpha^{3/2}} \right) \tag{6.2.9}$$

By multiplying this result with the inverse of the spacing along a ξ coordinate $(1/\sqrt{\gamma})$, the following expression is obtained

$$K_1 = \frac{y_\eta x_{\eta\eta} - x_\eta y_{\eta\eta}}{\sqrt{\gamma}\ \alpha^{3/2}} \tag{6.2.10}$$

which is the term that is subtracted from the adaptive grid equation for ξ in order to cancel the effects of curved η coordinate lines. An expression for the influence of curved ξ coordinates on the spacing along η coordinate lines can be obtained in a similar manner and is

$$K_2 = \frac{y_\xi x_{\xi\xi} - x_\xi y_{\xi\xi}}{\sqrt{\alpha}\ \gamma^{3/2}} \tag{6.2.11}$$

Thompson et al. [15] have derived similar expression, but do not implement them in the same way. It should be noted that in the computational plane, the terms K_1 and K_2 contain second derivatives of the variables x and y, and when eqs. (2.2.7) are transformed onto the computational plane, the second order central difference approximations to these terms depend explicitly on the point $(x_{i,j}, y_{i,j})$. However, in implementing the Newton-Raphson iterative solution procedure, the terms K_1 and K_2 are considered part of the source terms. Thus, the right hand side of eqs. (2.3.7) becomes

$$S_1 = \frac{\alpha J\ P_\xi}{P} - K_1 J^3$$

$$S_2 = \frac{\alpha J\ Q_\eta}{Q} - K_2 J^3 \qquad (6.2.12)$$

This is equivalent to neglecting the dependence of the finite difference approximation on the point $(x_{i,j}, y_{i,j})$ in obtaining eqs. (3.1.5) for the Newton-Raphson iterative procedure. The J^3 term in eqs. (6.2.12) appears in the transformation of eqs. (2.2.7) into the computational plane.

The adaptive grid generation equations, modified in this manner, will produce the grid shown in Figure 6.2b. In contrast to Figure 6.2a, it is clear that the effects of the curved boundaries have been successfully eliminated.

VI.4 Internal Grid Boundaries

One other problem that arose in the preliminary testing of the adaptive grid generation technique is the rapid upstream bending of the η coordinate lines emanating from the secant portion of the projectile surface. Figure 6.3a shows a grid network obtained using the adaptive grid generation equations with the curvature terms included and with λ = 0.5. As can be seen, the η coordinate lines emanating from the secant portion of the projectile quickly bend forward to fill the upstream region of the domain which results in a very skew grid. This characteristic of the grid does not appear as a problem until a solution using the Beam and Warming scheme is sought on such a grid. Figure 6.3b shows a time history of the solution obtained on such a grid for Mach 0.96 and as can be seen the pressure coefficient oscillates in time over

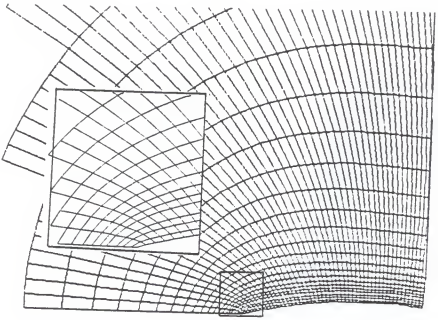


Figure 6.3a Grid network upstream of the projectile

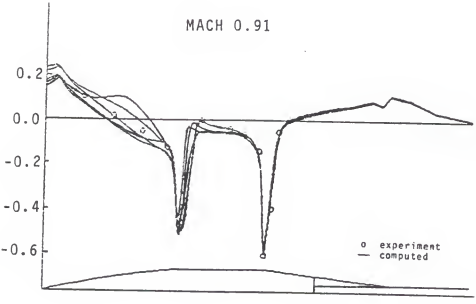


Figure 6.3b Lack of convergence over secant ogive with Beam and Warming scheme

the secant portion of the projectile. The solution did not converge and instead the oscillations

continued and eventually effected the solution downstream.

In order to prevent the upstream bending of the η coordinates, one η coordinate line in the initial grid network is designated as an internal boundary. This η coordinate line remains fixed in space during the iterativesolution procedure, thus no other η coordinate lines will pass through it. However, the points defining the internal boundary coordinate line are allowed to move along the coordinate so that they remain consistent with the spacing of the points along the adjacent η coordinate lines. This is accomplished by updating the arclength distribution along the internal boundary after each iterative sweep in which the new distribution is defined as the average of the arclength distributions along the two adjacent η coordinate lines. Figure 6.3c shows the grid obtained when the 12th η coordinate line from the projectile nose is designated as an internal boundary and as can be seen it prevents the η coordinate lines from bending upstream. It has been chosen to lie in a region in which no major grid clustering is expected so that it will not effect the adaptation of the grid network.

The length and number of points used along the ξ coordinate lines in each section may result in an abrupt change in the point spacing along the ξ coordinate lines passing through the boundary as can be seen in Figure 6.3c. Therefore the control function P, controlling the spacing along the ξ coordinate lines is locally modified near the internal boundary so that the grid point spacing varied smoothly through the internal boundary. This can be accomplished by requiring the normalized rate of change of grid point spacing along a ξ coordinate line ($s_{\xi\xi}/s_{\xi}$)

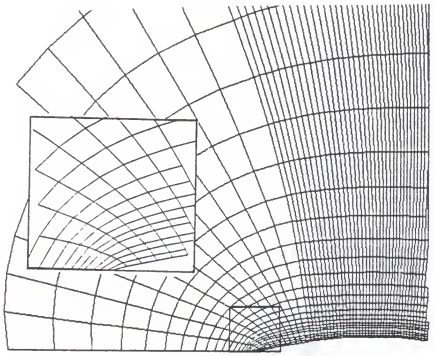


Figure 6.3c Grid network with internal boundaries

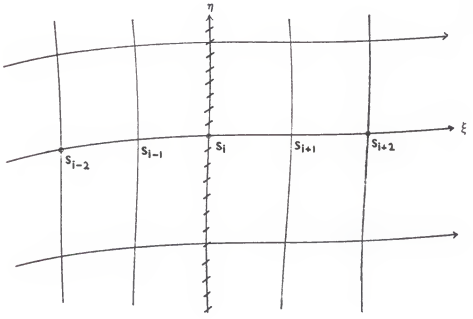


Figure 6.4 Grid points near internal boundary

be identical at the two points adjacent to the internal boundary. Referring to Figure 6.4, if the h coordinate line designated as an internal boundary contains the i^{th} point along a ξ coordinate line, we then require

$$\left(\frac{s_{\xi\xi}}{s_{\xi}} \right)_{i-1} = \left(\frac{s_{\xi\xi}}{s_{\xi}} \right)_{i+1} \tag{6.3.1}$$

To implement this requirement into the solution procedure, consider the one dimensional equation for adaptation along a ξ coordinate, eq. (2.3.8) which is written here as

$$\frac{P_{\xi}}{P} = \frac{s_{\xi\xi}}{s_{\xi}} \tag{6.3.2}$$

The left hand side of eq. (6.3.2) appears in the source terms of the two dimensional equations for adaptation, eqs. (2.3.7). Substitution of eq. (6.3.2) into the source terms for a grid point just ahead of the internal boundary results in the modified source terms at the point i-1

$$(S_i)_{i-1,j} = \alpha_{i-1,j} J_{i-1,j} \left(\frac{s_{\xi\xi}}{s_{\xi}} \right)_{i-1,j} \tag{6.3.3}$$

The modified source terms for the point just behind the internal boundary are obtained by replacing i-1 with i+1 in eq. (6.3.3). In order to apply the iterative solution technique a value for the terms $(s_{\xi\xi}/s_{\xi})$ in eqs. (6.3.3) for each point adjacent to the internal boundary will be needed. Their final values are not known a priori since they will depend on the final position of the points in each section, therefore they are set to

the current value

$$\left[\frac{s_{\xi\xi}}{s_{\xi}}\right]_{i-1,j} = \frac{(s_{i-2} - 2s_i + s_{i+2})}{(s_{i+2} - s_{i-1})/2} \qquad (6.3.4)$$

The grid that is obtained when using this technique is shown in Figure 6.5 in which it can be seen that the abrupt change in spacing across the internal boundary of the grid in Figure 5.3c has been reduced.

The use of internal boundaries provides a simple and effective way to control the grid points in the domain. In fact, it was found helpful to also designate the η coordinate line which separates the region over the sting from that over the projectile as an internal boundary to keep the η coordinate lines that originally began on the projectile surface from moving over the sting.

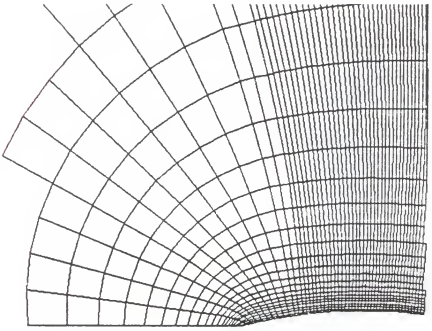


Figure 6.5 Smoothed grid spacing near the internal boundary

CHAPTER VII
THE SELF-ADAPTIVE COMPUTATIONAL METHOD

The adaptive grid generation technique can be incorporated naturally into the solution process of the flow solvers since the flow solver algorithms are time-marching. After a specified number of time steps, the grid network can be adapted to the control functions based on the current values of the flow variables. Thus the time-marching algorithm of the two flow solvers used here, the Thin Layer and TVD codes, proceeds as usual with intermittent calls to the Adaptive Grid code to update the grid network. In this approach, however, the relocation of the grid points in the physical domain due to the grid adaptation must be considered. The values of the flow variables at any time step are defined at the current grid point locations and any movement of the points will therefore distort the solution. This effect is critical for unsteady flow problems since the grid point movement will effect the solution accuracy. For steady flow problems, the grid point movement may effect the convergence rate and if it is large enough, it may cause the solution to diverge.

There are two basic approaches to account for the grid point movement resulting when adapting the grid network. The first is to view the grid network as a moving grid and include the time metrics in the coordinate transformation of the Navier-Stokes equations. These metrics can be written in the computational plane as

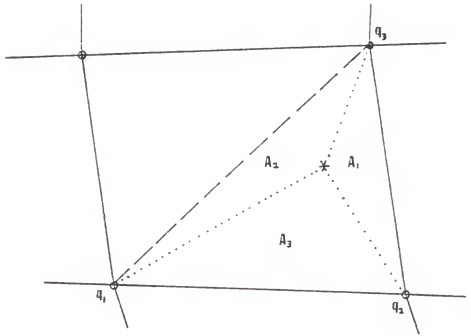
$$\xi_t = \xi_x x_\tau + \xi_y y_\tau$$
$$\eta_t = \eta_x x_\tau + \eta_y y_\tau \qquad (7.1)$$

The quantities x_τ and y_τ can be approximated by using a backward difference expression

$$x_\tau \simeq (x^{n+1}-x^n)/\Delta t$$
$$y_\tau \simeq (y^{n+1}-y^n)/\Delta t \qquad (7.2)$$

two in which x^n and y^n are the grid point locations of the previous grid network and x^{n+1} are y^{n+1} are the values of the adapted grid network. This approach requires the grid to be adapted every time step which is inefficient for steady flow problems. Furthermore, it has been shown that special procedures must be used in evaluating the time derivative of the Jacobian [20] in order to maintain the conservative property of the finite difference schemes.

Another approach, used here, is to interpolate the flow variables from the previous grid network onto the adapted grid network. This approach can be applied to both steady and unsteady flow problems and does not require the grid to be adapted every time step. Thus this approach can be used for both steady and unsteady flow problems. For the two- dimensional grid networks considered here, a linear interpolation is currently used. Each grid cell in the previous grid network is divided into two triangles. Once a grid point in the adapted grid network is located within a triangle of the previous grid network, the value of the flow variables at that point can be determined by interpolation. Referring to Figure 7.1, the value of a flow variable q at the point in the adapted grid network can be determined from the values of the flow



O previous grid network

* current grid network

Figure 7.1 Flow variable interpolation

variables at the vertices of the triangle as

$$q = \frac{(q_1A_1 + q_2A_2 + q_3A_3)}{(A_1 + A_2 + A_3)} \tag{7.3}$$

where A_1, A_2 and A_3 are the areas of the triangles formed with the grid point in the adapted grid network.

The basic algorithm for incorporating the adaptive grid generation technique into both the Thin Layer code and the TVD code consists of the following steps:

1. Input an initial grid network into the flow solver code and begin the time-marching algorithm.
2. Advance the solution a specified number of time steps, and submit the current grid network to the Adaptive Grid code.
3. Obtain an adapted grid network using the Adaptive Grid code.
4. Interpolate the flow variables onto the adapted grid network which now becomes the current grid network.
5. Repeat steps 2 through 5 until the solution converges for steady flow problems or repeat the steps until a specified time is reached for unsteady flow problems.

This process which couples the flow solver and the adaptive grid generation technique is referred to here as the self-adaptive computational technique. For the two problems considered here, the axisymmetric transonic flow over a projectile with sting and the transonic projectile base flow problem, only steady flow solutions are sought. Since the adapted grid network depends on the flow variables via the control functions, the convergence of the solution implies also that the grid network ceases to change with time. In all the examples of the following chapters the grid networks of Figures 5.4 and 5.10 are used as

the initial grid.

For the purpose of adapting the grid network, the source terms corrected to eliminate curvature effects, eqs. (6.2.12) were used in the adaptive grid generation equations. In solving the projectile problem with sting two internal η coordinates were designated as internal boundaries. The first is over the secant ogive as shown in Figure 6.5, the second separates the grid network over the projectile from that over the sting and is located at approximately x=7.0. In solving the projectile base flow problem, no internal boundaries were used.

CHAPTER VIII
RESULTS AND DISCUSSION

VIII.1 Choices for the Control Functions

In applying the self-adaptive computational technique it is necessary to define the control functions for adaptation. For the transonic projectile problem considered here both shocks and expansion waves will occur approximately normal to the projectile surface and require refined grid point spacing in the ξ coordinate direction for adequate resolution. The original choice for the function f in the control function P to obtain this clustering was the pressure gradient. However, it was found in the numerical experiments that the expansion waves occurring at the secant ogive/cylinder and cylinder/boattail junctures required smaller spacing than the shocks for good resolution. The pressure gradient was largest in the shocks and thus smaller spacing occurred in the shock regions rather at the expansion waves. A better choice for the control function was found to be the curvature of the pressure distribution along the streamwise coordinate direction,

$$f = \frac{\left| \frac{\partial^2 p}{\partial s^2} \right|}{\left(1 + \left(\frac{\partial p}{\partial s} \right)^2 \right)^{3/2}} \qquad (8.1.1)$$

where the derivatives are with respect to the arclength s along each ξ coordinate line. This function is largest at the expansion waves where the pressure gradient changed sign. At the base and peak of a shock eq. (8.1.1) also increased but it is not as large as the value near the expansion waves. Thus the smallest spacing will occur near the expansions and yet the grid points will also cluster, to a lesser extent, in the vicinity of shocks.

In the normal direction, the important area requiring increased resolution is the boundary layer and the logical choice for the function g in the control function Q is the velocity gradient. However, it was found that the control function based on the velocity gradient resulted in too many points being placed in the boundary layer and led thus to a rapid stretching of the grid point spacing along the η coordinate lines. Another choice for the function g, the exponential of the velocity gradient

$$g = \exp \left| \frac{\partial u}{\partial s} \right| \tag{8.1.2}$$

where u is the velocity magnitude and s measures along the η coordinate line, was found to yield better results. Figure 8.1 shows a plot of the control function Q along a typical S_η coordinate line. The control function resulting from the exponential of the velocity gradient is similar to the control function corresponding to the grid point distribution in the fixed grid networks which is known to give good results provided enough points are used. The control function obtained using the velocity gradient however is large for many grid points and,

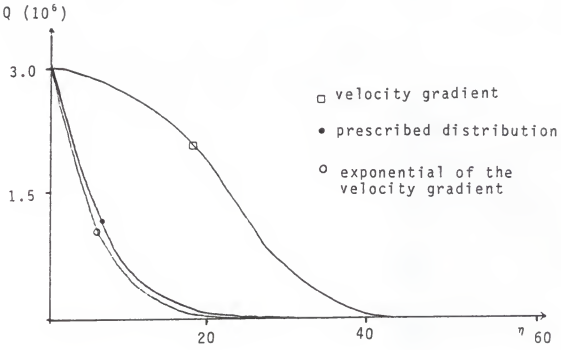


Figure 8.1 Comparison of control functions

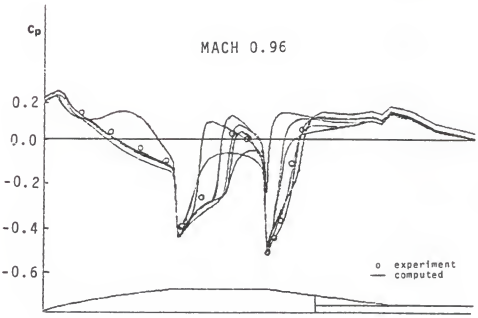


Figure 8.2 Lack of convergence using a self-adaptive computational technique with Beam and Warming scheme

thus, results in too many points being clustered into the boundary layer region at the expense of adequate resolution in other areas. The minimum and maximum grid point spacing along the η coordinate lines is 0.00002 and 6.0 respectively. In this direction it is important to have the small spacing at each point along the projectile surface in order to adequately resolve the viscous sublayer region. The parameter σ of eq. (4.1.4) was determined for each η coordinate line so that the small spacing would occur for each point on the projectile surface. However, in the ξ coordinate direction, the shocks and expansion waves are not expected to reach the outer boundary and the small spacing needed along the projectile surface is not required near that boundary. With the minimum and maximum grid point spacing prescribed as 0.02 and 0.22, respectively, σ is determined only for the ξ coordinate line coincident with the projectile surface and that value is used for all the remaining ξ coordinate lines. This had the effect of increasing the minimum spacing as the magnitude of the shocks and expansion waves diminished near the outer boundary.

VIII.2 Results Using the Self-Adaptive Computational Technique: The Beam and Warming Scheme

The first application of the self-adaptive computational technique is with the Beam and Warming scheme for the calculation of the projectile flow with sting. The Mach number is 0.96 and the Reynolds number is 76,000, a case for which experimental data is available. The time step used in the calculation is 0.1 and the explicit and implicit dissipation terms are 2 and 4 times the time step respectively. The initial grid network used is that of Figure 5.5 with IM=70 and JM=60. In the first

attempts to use the adaptive grid technique, the grid network was adapted every 200 time steps. A series of calculated pressure coefficients obtained with this approach are shown in Figure 8.2. As can be seen, the pressure coefficient oscillates widely over the projectile surface and shows no indication of converging. In a series of attempts to obtain better results, the time step was decreased to 0.05, the dissipation terms were doubled and the number of time steps between adaptation were both increased to 500 and reduced to 50. However, similar results were obtained in each case.

In the next case tried, the number of points in the streamwise direction was increased to 90 points, thus IM=90, and the grid network was adapted every time step. The results for this case are much better, however a converged solution was not obtained. Figure 8.3a shows the computed pressure coefficient at the times indicated, and as can be seen, the solution appears to have converged everywhere except in the shock boundary layer interaction regions. The shocks propagate upstream and downstream in a cyclic fashion. Figure 8.3b shows the same phenomenon occurring at a later time. In fact, the solution was continued for 8000 time steps and the propagation of the two shocks showed no indication of diminishing.

Figure 8.4a shows the reduced grid network which resulted at 5000 time steps. As is evident, the grid point spacing in the η coordinate direction is clustered near the projectile surface with a distribution similar to that of the hyperbolic grid in Figure 5.4. The spacing varies somewhat in the shock and expansion wave regions, most likely as a result of enforcing orthogonality. The clustering in the streamwise direction

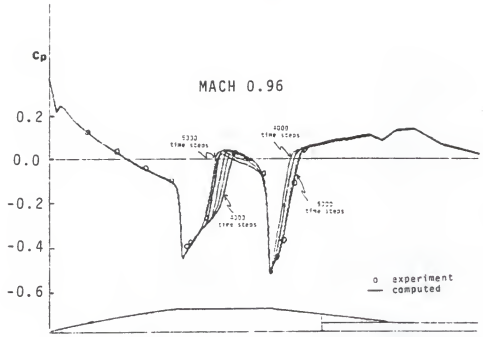


Figure 8.3a Computed solution between 4000 and 5000 time steps with Beam and Warming scheme

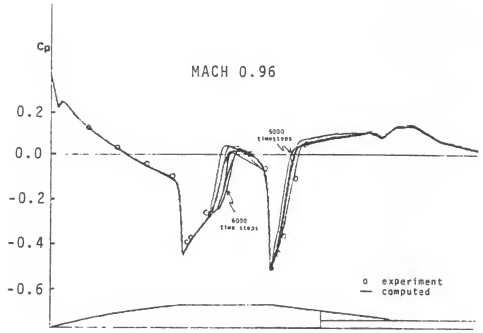


Figure 8.3b Computed solution between 5000 and 6000 time steps with Beam and Warming scheme

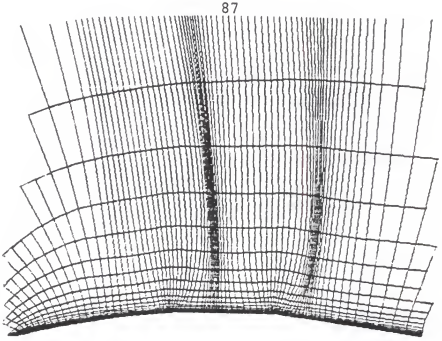


Figure 8.4a Adapted grid network at 5000 time steps (90 x 60)

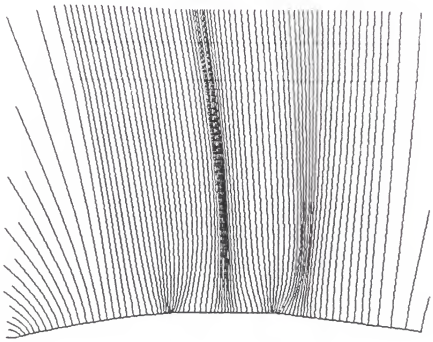


Figure 8.4b Adapted grid network at 6000 time steps (90 x 60)

clearly indicates a response to the control function P, which is based on the curvature of the pressure function. Clustering occurs at both junctures on the projectile surface and the adaptation to the shock structures on the cylinder and boattail are also quite evident. Note that the smallest spacing appears to be located in the shocks well above the projectile surface. This occurs because there is more competition for the small grid spacing on the projectile surface since both the shocks and expansion waves are strong in that region. Figure 8.4b shows the grid network corresponding to a later time. In this Figure and those following, the ξ coordinate lines are not shown in order to gain a better view of the grid spacing near the projectile surface. The grid configuration is basically the same as that of Figure 8.4a, however, the position of the shocks as indicated by the adaptation has changed. The shock on the cylinder has propagated downstream and the boattail shock has moved upstream. Referring to the shock positions evident in Figures 8.3a and 8.3b it can be seen the clustering in the grid network is successfully following the shock patterns.

In an attempt to improve these results the time step was reduced to 0.05 and the dissipation terms were again increased, however, similar results were obtained. Also the convergence criterion δ of eq. (3.1.12) used in the iterative solution procedure for the grid generation equation was varied. At the value of δ=0.005, only one iteration was required for the grid network to converge. The rapid convergence occurred because the grid was updated every time step and the solution, and consequently the control functions, did not change much between grid adaptations. When the criterion was reduced to δ=0.0025, 1 to 3 iterations were required for

the grid to converge at each time step, however it had no effect on the cyclic propagation of the shocks. Although a converged solution was not obtained using the adaptive grid generation technique with the Beam and Warming scheme, there are still some encouraging results with respect to the adaptive grid generation procedure. The maximum and minimum grid point spacing occurring along the projectile surface in the adapted grid network were within 5 percent of their prescribed values. The average minimum spacing along all the η coordinate lines was within 3 percent of the prescribed value of 0.00002 with a maximum deviation of 10 percent. The adapted grid networks shown in Figures 8.4a and 8.4b show that the adapted grid generation equations are capable of responding reliably and predictably to the chosen control functions and yield the prescribed maximum and minimum grid point spacing.

The adaptive grid generation equations, thus, have been shown to provide a well adapted grid network for the transonic flow problems provided that a good choice is made for the control functions. However, the convergence of the solution was effected by the use of the adaptive grid generation procedure with the Beam and Warming scheme. One reason for the failure of the solution to converge may be due to interpolation. Error introduced due to interpolating the flow variables onto each successive adapted grid network may continuously perturb the solution, causing it to oscillate. Another cause of this phenomenon may be the form of the dissipation terms in the Beam and Warming scheme. The added dissipation terms contain derivatives with respect to the computational coordinates and were added to the scheme after the governing equations were transformed. Since these terms are a function of the transformed

coordinates any change in the grid network used will change the dissipation terms and consequently alter slightly the transformed governing equations. Thus the continuous adaptation of the grid network continuously alters the dissipation terms and may be the reason the solution does not converge.

VIII.3 Results Using the Self-Adaptive Computational Technique: The TVD Scheme

Since the solutions obtained using the Beam and Warming scheme did not converge, another scheme, the TVD scheme was also used. Recall that the primary difference between the two schemes is the dissipation terms added to the governing equations. The initial grid network used is that of Figure 5.4 with IM=70 and JM=60. The time step used is 0.1 and the grid was adapted every 200 time steps. For Mach 0.96, the solution converged easily within 2000 time steps. Only one iteration was needed for the grid network to converge at 1800 and 2000 time steps which also indicates a converged solution. Figure 8.5a shows the computed pressure coefficient using the adapted grid generation procedure compared to the solution obtained on the fixed grid network of Figure 5.4. The solutions agree well with experimental data over most of the projectile surface, except for the predicted position of the shock over the cylinder. The position predicted using the adaptive grid generation technique is centered on the cylinder whereas the position predicted using the fixed grid network is slightly upstream of the center. Figure 8.5b shows a shadowgraph for corresponding flow conditions in which it can be seen that the experimentally predicted shock location is centered over the cylinder portion of the projectile. Thus, the use of the adaptive grid generation technique improved the solution accuracy. Figure 8.5c shows

the adapted grid network corresponding to the converged solution. Again, clustering near the secant ogive/cylinder and cylinder/boattail junctures is evident as well as the adaptation to the shocks. In comparison to the fixed grid network of Figure 5.4 it is evident that the grid spacing along the projectile surface is refined at the center of the cylinder portion due to the adaptation and is responsible for the increased solution accuracy.

Two other cases were run using the identical procedure used for the case of Mach 0.96. Figure 8.6a shows a comparison of the solution obtained using both the fixed grid network and the adaptive grid generation technique for Mach 0.91, another case for which experimental data is available. The only improvement obtained using the adaptive grid generation procedure is an increase in the peak of the shock occurring over the cylinder portion of the projectile. However, the peak still does not reach the experimentally predicted value. In an attempt to increase the accuracy in that region, 10 more points were added in the ξ coordinate direction, thus IM=80, however no further improvement in the solution obtained with the adaptive grid generation technique was evident. The results thus may be the best that the TVD scheme can yield given the current grid configuration and the shortened pressure peak is probably due to the larger dissipative error associated with the TVD scheme. Figure 8.6b shows the grid network corresponding to the converged solution. Again grid point clustering in response to the curvature of the pressure is evident, and as can be seen by the adaptation of the grid network, both shocks occur directly downstream of the expansion waves.

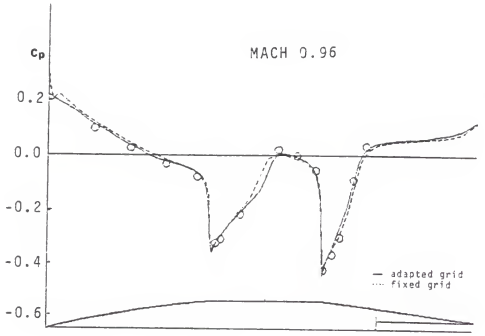


Figure 8.5a Comparison of computed solutions on fixed and adaptive grid networks with TVD scheme

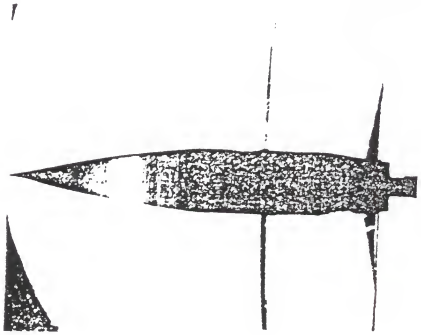


Figure 8.5b SOCBT shadowgraph for Mach 0.96 (reprinted from reference 24)

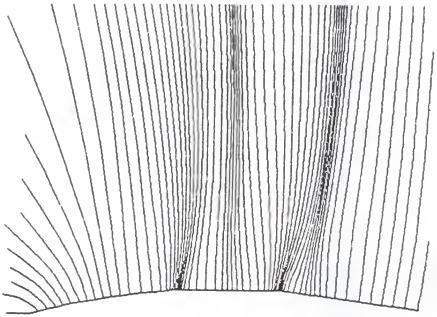


Figure 8.5c Adapted grid network for converged solution for Mach 0.96

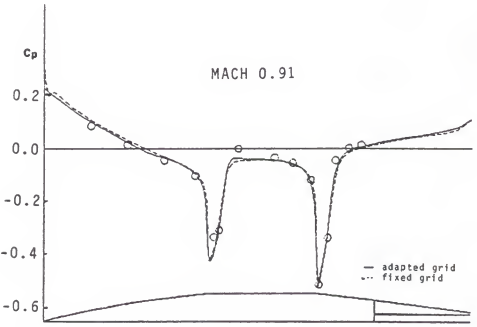


Figure 8.6a Comparison of computed solutions on fixed and adapted grid networks with TVD scheme

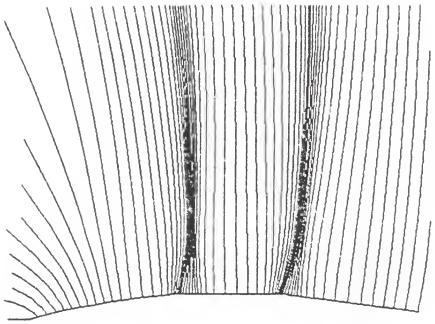


Figure 8.6b Adapted grid network for converged solution for Mach 0.91

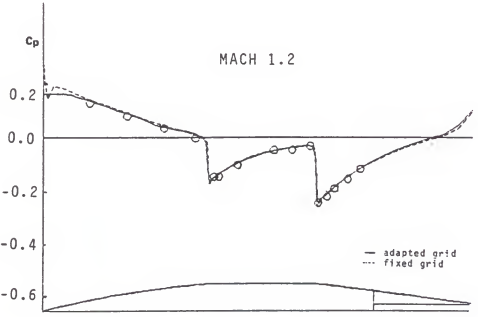


Figure 8.7a Comparison of computed solutions on fixed and adapted grid networks with TVD scheme

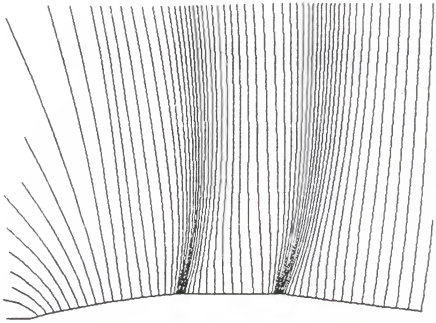


Figure 8.7b Adapted grid network for converged solution for Mach 1.2

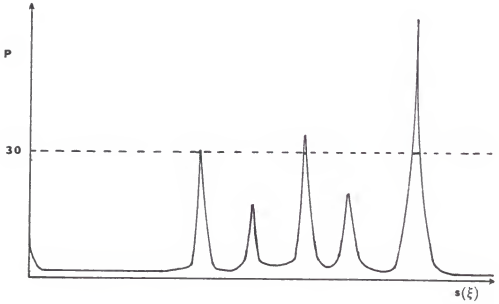


Figure 8.8 Setting maximum value for the control function

The results for a third case, Mach 1.2, are shown in Figures 8.7a and 8.7b. There is no noticeable difference between the computed results on the fixed grid network and the results obtained using the adaptive grid generation procedure except near the projectile nose where the grid configuration differs the most. In the adapted grid network, clustering again is evident near the junctures, however no shocks are present for this case, a result also evident in the grid network. The equal accuracy of the solutions obtained with the adapted grid generation procedure and on the fixed grid network is a result of the similarity of the grid networks. In comparing the adapted grid networks of Figures 8.6b and 8.7b to the fixed grid network of Figure 5.4 it is evident that they have basically the same features in that the grid point clustering along the ξ coordinates is concentrated near the two junctures. Thus the original fixed grid network is well adapted to the problem since the shocks for case of Mach 0.91 are so close to the expansion waves and no shocks occur in the case of Mach 1.2. However, it should be noted that the adaptive grid generation procedure successfully produced a good adapted grid network without any knowledge of the position or orientation of the important flow phenomenon. The choice and construction of the proper control functions of course are important in obtaining these results. In the context of these results it can be concluded that the adaptive grid generation procedure can reliably produce a good adapted grid network provided good choices are made for the control functions.

VIII.4 Application to the Projectile Base Flow Problem

In a final case, the self-adaptive computational technique was applied to the projectile base flow problem. The flow conditions were Mach 0.96 and a Reynolds number of 76,000. As in the applications to the projectile with sting, the time step used in the TVD scheme is 0.1 and the grid is adapted every 200 time steps. The grid network of Figure 5.10 was used as the initial grid, however in the adaptive grid procedure no internal boundaries were used. The η coordinate lines emanating from the boattail are expected to bend downstream in the current configuration and, thus, should prevent the lines emanating from the secant ogive from bending to far foward.

Before applying the self-adaptive computational technique to compute the solution, the 70x60 grid network was adapted to the converged solution obtained on the 70x60 fixed grid network. It was found that the control function in a small region near the base corner was so large that most of the grid points clustered in that region and resulted in poor resolution near the shocks and expansion waves. The method used to alleviate this problem, is to prescribe a maximum allowable value for the curvature of the pressure distribution. Any value of the curvature above this value was simply set to the prescribed maximum value. This approach is depicted in Figure 8.8. It introduces another parameter but results in a much better grid network and makes the self-adaptive grid generation much more reliable. Currently the maximum value used is 30, which was determined after some experimentation.

The calculated pressure coefficent obtained using the self-adaptive computational technique is shown in Figure 8.9a; the distribution over

the base region is shown in Figure 8.9b. There is generally good agreement between the two calculated solutions with the experimental data over the projectile surface with a slight discrepency near the shock on the cylinder. At the base corner and along the base the two calculated solutions are different though their trends are similar. Without any experimental data for comparison it is difficult to judge the accuracy. Furthermore, the thin layer approximation and the current turbulence model may not be valid near the base since a separated wake region exists.

The adaptive grid network corresponding to the converged solution is shown in Figure 8.10a and an enlarged view of the grid network near the base corner is shown in Figure 8.10b. In comparing this grid network to that of Figure 8.5c for the projectile with sting problem it is evident that the η coordinate lines emanating from the front portion of the projectile do bend upstream, but not so drastically, and do not prevent the convergence of the solution using the TVD scheme. There is the same general adaptation to the expansion waves and shocks, however the adaptation to the shock over the boattail appears to bend downstream, a result due to the smoothing of the control functions in the computational plane. Adaptation of the grid network near the base corner region is also evident. Note that the grid lines remained nearly orthogonal in the base region even though the normal direction changes direction rapidly near the sharp corner. In future work, the orthogonlity near the corner may be improved by decreasing the spacing of the ξ coordinate lines in the reduced grid network. Also, the grid spacing between the η coordinate lines is fairly uniform and smooth and the coordinate lines did not

overlap at the corner, a result attributed to the elliptic nature of the adaptive grid generation equations. The adaptive grid generation procedure, thus, can successfully provide a good adapted grid for this complex geometry.

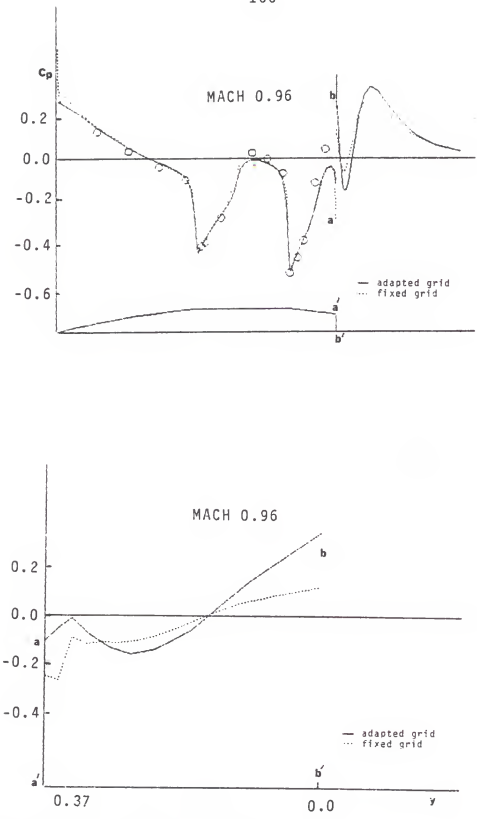


Figure 8.9 Comparison of computed solutions for the projectile base flow problem with TVD scheme

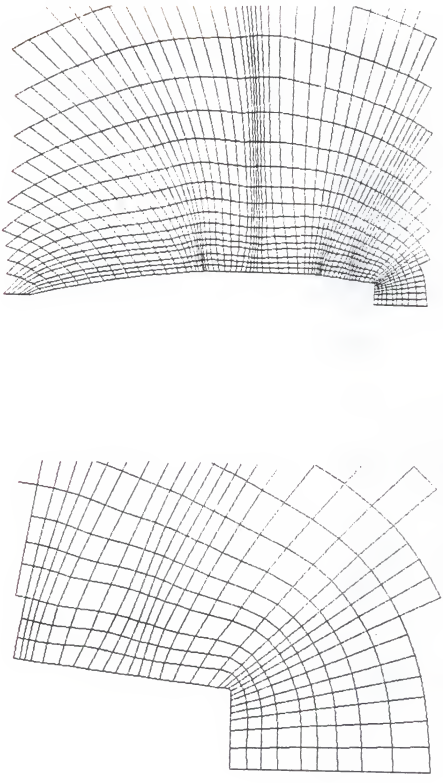


Figure 8.10 Adapted grid network for the converged solution for Mach 0.96

CHAPTER IX
CONCLUDING REMARKS

The self-adaptive computational technique developed here has been shown to provide good adaptive grid networks for the transonic projectile flow problems considered. The equations governing the grid adaptation were developed, based on a variational approach, to provide adaptation independently in each coordinate direction and also enhance both grid orthogonality and smoothness. The resulting set of elliptic partial differential equations provide explicit control of orthogonality and adaptation while smoothness is inherent in the ellipticity of the governing equations. Smoothness can also be controlled in the definition of the control functions.

A series of modifications were made to improve both the adaptive grid generation equations and their solution procedure. Local scaling of the equations was found useful for highly stretched meshes, and an effective technique to eliminate the effects of curved boundaries, especially useful in the projectile base flow problem, was developed. Also, the numerical solution procedure for the adaptive grid generation equations was improved to increase the poor convergence rate due to the high aspect ratio grid cells present in the grid networks used for transonic viscous flow problems. Finally, the use of internal grid boundaries, which were used in the projectile flow problem with sting, helped to provide a better grid network by restricting the movement of the grid points.

In coupling the adaptive grid generation procedure with the flow solvers it was shown that the adaptive grid generation equations could provide good adapted grid networks for the transonic projectile flow problems. It was found that the control function based on the curvature of the pressure distribution was more effective than the pressure gradient when both shocks and expansion waves are present. Also, in using a control function based on the velocity gradient, it was shown that the adaptive grid generation procedure could provide the extremely refined grid point spacing necessary in the boundary layer region.

The most important feature of the adaptive grid generation procedure developed here is the simple relationship between the control functions and the grid point spacing which is maintained in an elliptic set of partial differential equations. As more complex problems are attempted, the control functions for each coordinate direction can be carefully designed, based on various physical gradients, prescribed grid point distributions and combinations thereof, to obtain the desired adaptation. The elliptic nature of the adaptive grid generation equations will be helpful, especially for complex geometries, in maintaining a one to one mapping and preventing highly distorted grid networks from occuring.

APPENDIX

$$\xi_{xx} = (-y_\eta \ L_{yy}(x) + x_\eta L_{yy}(y))/J^3$$

$$\xi_{xy} = (-y_\eta \ L_{xy}(x) + x_\eta L_{xy}(y))/J^3$$

$$\xi_{yy} = (-y_\eta \ L_{xx}(x) + x_\eta L_{xx}(y))/J^3$$

$$\eta_{xx} = (-y_\xi \ L_{yy}(x) + x_\xi L_{yy}(y))/J^3$$

$$\eta_{xy} = (-y_\xi \ L_{xy}(x) + x_\xi L_{xy}(y))/J^3$$

$$\eta_{xx} = (-y_\xi \ L_{xx}(x) + x_\xi L_{xx}(y))/J^3$$

$$L_{xx} = x_\eta^2 \frac{\partial^2}{\partial \xi^2} - 2x_\eta x_\xi \frac{\partial^2}{\partial \xi \partial \eta} + x_\xi^2 \frac{\partial^2}{\partial \eta^2}$$

$$L_{xy} = x_\eta y_\eta \frac{\partial^2}{\partial \xi^2} - (x_\eta y_\xi + z_\xi y_\eta) \frac{\partial^2}{\partial \xi \partial \eta} + x_\xi y_\xi \frac{\partial^2}{\partial \eta^2}$$

$$L_{yy} = y_\eta^2 \frac{\partial^2}{\partial \xi^2} - 2y_\eta y_\xi \frac{\partial^2}{\partial \xi \partial \eta} + y_\xi^2 \frac{\partial^2}{\partial \eta^2}$$

$$J = x_\xi y_\eta - x_\eta y_\xi$$

REFERENCES

1. Kutler, P, "A Perspective of Theoretical and Applied Computational Fluid Dynamics," AIAA Journal, Vol. 23, NO. 3, pp. 328-341, 1984.

2. Thompson, J.F., Thames, F.C., and Mastin, C.W., "Automated Numerical Generation of Body-Fitted Curvilinear Coordinate System for Field Containing Any Number of Arbitrary Two-Dimensional Bodies," J. Comp. Phys., Vol. 15, pp. 299-319, 1974.

3. Mastin, C.W., "Error Induced By Coordinate Systems," Numerical Grid Generation, Ed. J.F. Thompson, North-Holland, p. 41, 1981.

4. Hsu, C.C., and Chang, T.H., "On a Numerical Solution of Incompressible Turbulent Boundary Layer Flow," Finite Element Flow Analysis, Ed. T. Kawai, Univ. of Tokyo Press, Tokyo, pp 219-226, 1982.

5. Saltzman, J. and Brackbill, J.U., "Application and Generalization of Variational Methods for Generating Adaptive Meshes," Numerical Grid Generation, Ed. J.F. Thompson, North-Holland, pp. 865-884, 1981.

6. Hsu, C.C. and Tu, C.G., "An Adaptive Grid Generation Technique Based on Variational Principals for Transonic Projectile Aerodynamic Calculation," To Appear in Int. J. Numerical Methods in Fluids.

7. Nakahashi, K., and Deiwert, G.S., "A Self-Adaptive Grid Method with Applications to Airfoil FLow," AIAA Paper 85-1525.

8. Dywer, H.A., Smooke, M.D. and Kee, R.J., "Adaptive Gridding for Finite Difference Solutions to Heat and Mass Transfer Problems," Numerical Grid Generation, Ed. J.F. Thompson, North-Holland, New York, p. 339, 1982.

9. Gnoffo, P. A., "A Vectorized, Finite Volume, Adaptive-Grid algorithm for Navier-Stokes Calculations," Numerical Grid Generation, Ed. J. F. Thompson, North-Holland, New York, p. 819, 1981.

10. Thompson, J.F., "A Survey of Dynamically-Adaptive Grids In the Numerical Solution of Partial Differential Equations," Applied Numerical Mathematics 1, North-Holland, New York, 1985.

11. Rai, M.M. and Anderson, D.A., "Grid Evolution in Time Asymptotic Problems," J. of Comp. Phys., Vol 43, pp. 327-344, 1981.

12. Ames, W.F., "Numerical Methods for Partial Differential Equations," Academic Press, Inc., New York, 1977.

13. Tu, C.G., "Development of An Adaptive Grid Generation Code for Projectile Aerodynamic Computation," Ph.D. Dissertation, University of Florida, 1985.

14. Pierson, B.L. and Kutler, P., "Optimal Nodal Point Distribution for Improved Accuracy in Computational Fluid Dynamics," AIAA Journal, Vol. 18, NO. 1, pp. 49-54, 1980.

15. Thompson, J. F., Warsi, Z. U. A. and Mastin, C. W. Numerical Grid Generation: Foundations and Applications, North-Holland, New York, pp. 215-221, 1985.

16. Nietubicz, C.J., Heavey, K.R. and Steger, J. L., "Grid Generation Techniques for Projectile Configurations," Proc. 1982 Army Numerical Analysis and Computers Conference, ARO Rept. 82-3, February 1982.

17. Warming, R.F. and Beam, R., "On the Construction and Application of Implicit Factored Schemes for Conservation Laws," SIAM-AMS Proceedings, Vol. 2, pp. 85-99, 1978.

18. Pulliam, T.H., and Steger, J.L., "Implicit Finite-Difference Simulations of Three-Dimensional Compressible Flow," AIAA Journal, Vol. 18, NO. 2, pp. 167-169, 1980.

19. Nietubicz, C.J., Pulliam, T.H. and Steger, J.L., "Numerical Solutions of the Azimuthal Invariant Thin Layer Navier-Stokes Equations," AIAA Paper 79-0010, Presented at AIAA 17th Aerospace Sciences Meeting, New Orleans, Louisiana.

20. Anderson, D.A., Tannehill, J.C. and Pletcher, R.H., Computational Fluid Mechanics and Heat Transfer, McGraw-Hill, New York, p. 546, 1984.

21. Baldwin, B.S., and Lomax, H., "Thin Layer Approximation and Algebraic Model for Separated Turbulent Flows," AIAA Paper 78-257, Presented at AIAA 16th Aerospace Sciences Meeting, Huntsville, Alabama, 1978.

22. Yee, H.C., "Linearized Form of Implicit TVD Schemes for the Multidimensional Euler and Navier-Stokes Equations," Comp. & Math. with Appls., Vol. 12a, Nos. 4/5, pp. 413-412, 1986.

23. Shiau, N. and Hsu, C.C., "A Diagonalized TVD Scheme for Turbulent Transonic Projectile Aerodynamic Calulations," submitted to 8th Int. Conf. on Computing Methods in Applied Sciences and Engineering, Versailles, France.

24. Sahu, J., Nietubicz, C.J. and Steger, J.L., "Numerical Computation of Base Flow for a Projectile at Transonic Speeds," ARBRL-TR-02459, U.S. Army Ballistic Research Laboratory, June 1983.

25. Kayser, L.D. and Whiton, F., "Surface Pressure Measurements on a Boattailed Projectile Shape at Transonic Speeds," ARBRL-MR-03161, U.S. Army Ballistic Research Laboratory, March, 1982.

BIOGRAPHICAL SKETCH

Christopher William Reed was born in Rahway, New Jersey, in 1960. He graduated from the Bolles School in Jacksonville, Florida, in 1978 and received a B.S. in engineering science and mechanics from the Georgia Institute of Technology. He earned his M.S. from the University of Florida Department of Engineering Sciences in 1984 and in 1986 was married to Shelley Anne Baylis. He received his Ph.D. from the University of Florida Department of Engineering Sciences in 1987.

I certify that I have read this study and that in my opinion it conforms to acceptable standards of scholarly presentation and is fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy.

Chen-Chi Hsu, Chairman
Professor of Engineering Sciences

I certify that I have read this study and that in my opinion it conforms to acceptable standards of scholarly presentation and is fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy.

Bernard Leadon
Professor of Engineering Sciences

I certify that I have read this study and that in my opinion it conforms to acceptable standards of scholarly presentation and is fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy.

Ulrich H. Kurzweg
Professor of Engineering Sciences

I certify that I have read this study and that in my opinion it conforms to acceptable standards of scholarly presentation and is fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy.

David W. Mikolaitis
Assistant Professor of Engineering Sciences

I certify that I have read this study and that in my opinion it conforms to acceptable standards of scholarly presentation and is fully adequate, in scope and quality, as a dissertation for the degree of Doctor of Philosophy.

David C. Wilson
Associate Professor of Mathematics

This dissertation was submitted to the Graduate Faculty of the College of Engineering and to the Graduate School and was accepted as partial fulfillment of the requirements for the degree of Doctor of Philosophy.

August 1987

Herbert A. Lewis
Dean, College of Engineering

Dean, Graduate School